JSC-12941
80-10170
NASA CR-
160680

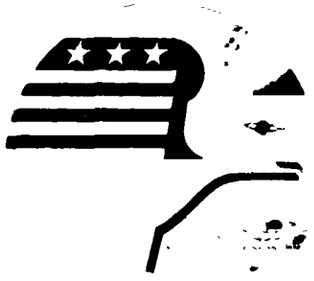
"AS-BUILT" DESIGN SPECIFICATION FOR THE YIELD ESTIMATION SUBSYSTEM (YES) OPERATIONAL ROBERTSON PHENOLOGICAL MODEL

Job Order 74-963
AD 63-1347-4963-11

(E80-10170) AS-BUILT DESIGN SPECIFICATION FOR THE YIELD ESTIMATION SUBSYSTEM (YES) OPERATIONAL ROBERTSON PHENOLOGICAL MODEL (Lockheed Electronics Co.) 118 p HC A06/MF A01 CSCL 02C G3/43 — N80-28771 Unclas 00170

Prepared By
Lockheed Electronics Company, Inc.
System and Services Division
Houston, Texas

Contract NAS 9-15200
For
EARTH OBSERVATIONS DIVISION
SPACE AND LIFE SCIENCES DIRECTORATE



National Aeronautics and Space Administration

LYNDON B. JOHNSON SPACE CENTER

Houston, Texas

June 1977

LEC-10743

JSC-12941

"AS-BUILT" DESIGN SPECIFICATION

FOR THE

YIELD ESTIMATION SUBSYSTEM (YES)

OPERATIONAL ROBERTSON PHENOLOGICAL

MODEL

Job Order 74-963
AD 63-1347-4963-11

PREPARED BY

K. Williams

APPROVED BY

P.D. Krumm, Supervisor
Software Development Section

Prepared By

Lockheed Electronics Company, Inc.

For

Earth Observations Division

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
LYNDON B. JOHNSON SPACE CENTER
HOUSTON, TEXAS

June 1977

CONTENTS

Section		Page
1.	SCOPE	1-1
2.	APPLICABLE DOCUMENTATION	2-1
3.	SYSTEM DESCRIPTION	3-1
	3.1 HARDWARE DESCRIPTION	3-2
	3.2 SOFTWARE DESCRIPTION	3-2
	3.2.1 TAPE AND DISK DATASETS AND ATTRIBUTES	3-7
	3.2.2 SUBROUTINE INIT	3-10
	3.2.3 SUBROUTINE SWAP	3-13
	3.2.4 SUBROUTINE TERM	3-20
	3.2.5 PROGRAM CLEAR	3-23
	3.2.6 JCL FILE CLEAR	3-26
	3.2.7 PROGRAM CAPTURE	3-27
	3.2.8 JCL FILE CAPTURE	3-34
	3.2.9 SUBROUTINE CVB	3-35
	3.2.10 PROGRAM PRINT	3-38
	3.2.11 JCL FILE LIST	3-41
	3.2.12 PROGRAM EDITOR	3-42
	3.2.13 JCL FILE EDIT	3-48
	3.2.14 SUBROUTINE CHECK	3-49
	3.2.15 SUBROUTINE REPLAC	3-59
	3.2.16 PROGRAM CROPCALN	3-67
	3.2.17 JCL FILE SUZYQ	3-74

Section Page

3.2.18 SUBROUTINE START 3-76

3.2.19 SUBROUTINE PHENO 3-79

3.2.20 FUNCTION DL 3-84

3.2.21 PROGRAM INSDEL 3-87

3.2.22 JCL FILE INSERT 3-93

3.2.23 PROGRAM DRECOVER 3-94

3.2.24 PROGRAM JSCTAPE 3-97

4. OPERATION

4.1 USER DOCUMENTATICN 4-1

4.2 PROCEDURES 4-1

4.2.1 CONSTRUCTING THE INDEX FILE (DRECOVER) . . . 4-2

4.2.2 CROP CALENDAR PROCEDURES 4-2

4.3 FLOWCHART 4-12

1. SCOPE

This document describes the Yield Estimation Subsystem (YES) operational implementation of Robertson's Phenological Model for small grains. Specifically, this document describes the software established to compute Robertson's biometeorological time scale for all LACIE weather stations from approximations to daily maximum and minimum temperatures. This document does not describe the software at NMC that extracts the weather data or the software at JSC that writes the AI crop calendar reports.

2. APPLICABLE DOCUMENTATION

1) Operating Procedures for the LACIE/YES Gridded Crop Calendar Report Writer by D.D. Wilcox. LEC-9955, Dec., 1976.

2) System Description of the LACIE/YES Gridded Crop Calendar Report Writer by D.D. Wilcox, LEC-10004, Jan., 1977.

3) AD 63-1347-4963-11 Documentation of Automated Crop Calendar Operation.

4) "A Biometeorological Time Scale for a Cereal Crop Involving Day and Night Temperatures and Photoperiod". G.W. Robertson, Intl. J. Biomet., 12#3 (191-223), (1968).

5) Level 3 Baseline
   Yield Estimation Subsystem (YES)
   Requirements
   LACIE C00200 Sept., 1976

6) Level 3 Baseline
   NOAA/NASA/USDA
   Interface Control Document
   LACIE C00710 Sept., 1976

7) A FORTRAN Implementation of the Robertson Phenological Model, D.D. Wilcox, G. Champagne, R. Baskett, S. Wooley. LEC Tech. Memo, LEC-5974.

3. SYSTEM DESCRIPTION

The Robertson Phenological Model is an algorithm for converting point estimates of daily maximum and minimum temperature and daylength to a point estimate of Robertson's Biometeorological Time Scale, a numerical scale describing the development of wheat. Knowledge of the stage of wheat development is useful to analyst interpreters (AI) in the estimation of wheat acreages. The Robertson Model has been implemented to make this knowledge available.

The Robertson Scale ranges continuously from 1.0, called planting, to 6.0, called ripe. The Robertson Model assigns to each integral range 1.0 to 2.0,..., 5.0 to 6.0, an equation $f_{1.0\text{-}2.0}(t_{max}, t_{min}, \text{daylength}), \ldots, f_{5.0\text{-}6.0}(t_{max}, t_{min}, \text{daylength})$. To advance the model one day, let $D_{yesterday}$ be the development yesterday. From $D_{yesterday}$ select one of $f_{1.0\text{-}2.0}, \ldots, f_{5.0\text{-}6.0}$, as $f_{yesterday}$. Obtain yesterday's t_{max}, t_{min}, and daylength, and compute:

$$D_{today} = D_{yesterday} + f_{yesterday}(t_{max}, t_{min}, \text{daylength}).$$

Clearly t_{max} and t_{min} are observations at the weather stations. It is most useful to have D_{today} at the sample segments (LACIE modelling region, for example). Therefore, some method of extending the weather station developments to the sample segments is necessary.

It should now be unequivocal that any implementation of the Robertson Model must have three major components:

1) A weather data collecting component,
2) a crop modelling component, and
3) a component that extends the output of (2) to the sample segments and writes a report.

This document describes the structure and operation of component 2. Component 3 is described in documents 1 and 2 of section 2.0 of this report. Component 1 is in the implementation stage. However, some software is described here which performs data collection and editing. It is anticipated these functions of component 2 will be abandoned when the final version of component 1 is put to use. From this point, component 2 will be refered to as 'the system'.

3.1 HARDWARE DESCRIPTION

These programs and datasets are resident on the IBM 360/195 complex at Suitland, Maryland. They should be transferable to any IBM 360-370 series machine with sufficient disk to handle the datasets and main memory to support the PL/I optomizing compiler.

3.2 SOFTWARE DESCRIPTION

The software consists of eight programs: one to prepare the station index file, six to prepare the crop calendar output, and one to copy this output from disk to tape. The program to prepare the station index file, DRECOVER, is input by cards at CCEA/Columbia. The six crop calendar preparation source programs are kept on cards at CCEA/Columbia and are listed in executable form on the partitioned dataset W.EDS. CCEA.PHASE3.LOAD; the executable programs are called by the following JCL files maintained on card decks at CCEA/Columbia:

1) CLEAR, used to initialize disk file CROPDATA,
2) CAPTURE, used to build the file,
3) LIST, which lists CROPDATA and can be accessed at any time during preparation of the file,
4) EDIT, which edits the file for mistakes and missing data,
5) SUZYQ, which performs crop modelling, and
6) INSERT, used to enter corrections into CROPDATA.

JSCTAPE, the program copying the crop calendar output to tape, is input by cards at CCEA/Columbia each time it is executed.

In general, a crop calendar update will require CLEAR, followed by several runs of CAPTURE, followed by LIST. At this point, EDIT is run to correct for missing data and mistakes. SUZYQ is run producing updates; any errors not corrected by EDIT are detected by SUZYQ. INSERT is then used to complete editing of the weather dataset and SUZYQ is rerun. After the final run of SUZYQ, JSCTAPE is run to produce a tape for transmittal to JSC.

Two (logical) files are maintained by SUZYQ to record the last predicted crop stage at each update, these files being in an Old Master/New Master relationship. In this way an error in

the preparation of the data (ex.,incorrect data captured, not enough data captured) can be handled by going back to the Old Master and beginning the process anew. See Section 3.2.1.6 for further discussion. Data flow and file usage are shown in Figure 3-1.

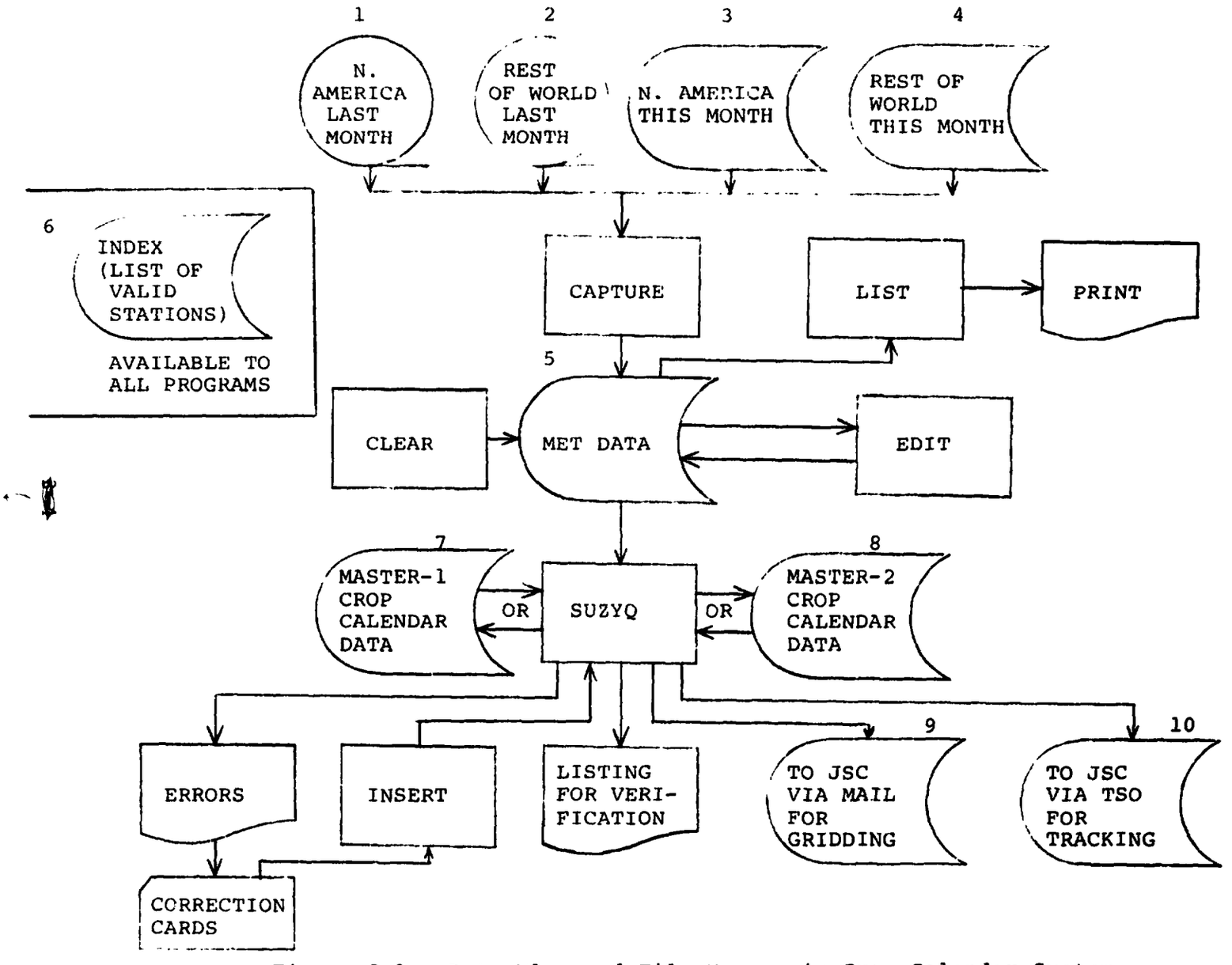


Figure 3-1.— Data Flow and File Useage in Crop Calendar System

Figure 3-1 (continued)

DATASET NAMES AND FLOWCHART REFERENCES

Flowchart Box #	DSN
1	CCEA.xxx.SRTD (see Section 3.2.1.1)
2	CCEA.DAILY.xxx (see 3.2.1.2)
3	W.EDS.CCEA.DAILY (see 3.2.1.3)
4	W.EDS.CCEA.DAYDATA (see 3.2.1.4)
5	W.EDS.CCEA.GROUPi.CROPDATA (see 3.2.1.7)
6	W.EDS.CCEA.GROUPi.INDEX (see 3.2.1.5)
7	W.EDS.CCEA.GROUPi.MASTER1 (see 3.2.1.6)
8	W.EDS.CCEA.GROUPi.MASTER2 (see 3.2.1.6)
9	W.EDS.CCEA.GROUPi.GRIDINFO (see 3.2.1.8)
10	W.EDS.CCEA.GROUPi.ARNO (see 3.2.1.9)

xxx = month of interest, 3 letter abbreviation; ex. JAN,FEB,MAR

i = I for Group I countries(U.S.A., INDIA, CHINA)
= II for Group II countries(U.S.S.R., CANADA)

The software components are divided into four classes:

1) Weather Data Access Routines:
   These are a set of routines, INIT, SWAP, and TERM which provide a uniform method of reading and writing data to the weather data files.
2) File Manipulation Routines:
   These routines include the main programs CLEAR, which initializes the weather data file, CAPTURE, which builds it, EDITOR, which edits out most mistakes and missing values, INSDEL, which makes card-input changes in the file, and PRINT which lists it.

   Also included is the program DRECOVER which sets up the INDEX file, the supporting routine CVB located in the PDS W.EDS.CCEA.PROD.LOAD and called by CAPTURE, and the JCL files which are needed for invocation of the main programs.

   The following is a list of the JCL files and their corresponding main programs:

File Name	Program
CLEAR	CLEAR
CAPTURE	CAPTURE
EDIT	EDITOR
INSERT	INSDEL
LIST	PRINT

3) Crop Modelling Routine:
   This includes the main program CROPCALN, supporting routines START and PHENO, function DL, and the JCL file SUZYQ, necessary for invocation.
4) Output Tape to JSC:
   This is a deck of JCL and the program JSCTAPE that produces the interface tape for shipment to JSC.

3.2.1 TAPE AND DISK DATASETS AND ATTRIBUTES

The following datasets will be referred to in subsequent descriptions of programs and subroutines.

3.2.1.1 CCEA.xxx.SRTD

North American data for past months on tape, where'xxx'is the month abbreviation (ex., Jan. Feb., etc.)

```
RECFM = FB                    UNIT = TAPE 9
LRECL = 560                   VOL = Eyyyyy, where 'yyyyy'
BLKSIZE = 5600                      is the tape number
```

This dataset is accessed through the CAPTURE routine; a separate tape exists at Suitland for each month of past data and is differentiated from the others by both the DSN and the tape number.

3.2.1.2 CCEA.DAILY.xxx

Non-North American data for past months on tape, where 'xxx' is the month abbreviation.

```
LRECL = 384
BLKSIZE = 12672
```

All other information is identical to CCEA.xxx.SRTD discussed in 3.2.1.1.

3.2.1.3 W.EDS.CCEA.DAILY

North American temperature data for the current month on disk.

```
RECFM = FB
LRECL = 560
BLKSIZE = 5600
```

This dataset is accessed during CAPTURE.

3.2.1.4 W.EDS.CCEA.DAYDATA

Non-North American temperature data for the current month on disk.

```
CRECL = 384
BLKSIZE = 12672
```

All other information is identical to W.EDS.CCEA.DAILY discussed in 3.2.1.3.

3.2.1.5 W.EDS.CCEA.{GROUP I / GROUP II}.INDEX

RECFM = FB
LRECL = 1024
BLKSIZE = 10240
DSORG = PS

The INDEX file for each group contains the number of stations on the file, followed by a list of station World Meteorological Organization (WMO) numbers and each station's three closest neighbor stations.

INDEX is created by the program DRECOVER.

3.2.1.6 W.EDS.CCEA.{GROUP I / GROUP II}.{MASTER 1 / MASTER 2}

RECFM = FB
LRECL = 80
BLKSIZE = 12960
DSORG = PS

Each crop calendar group must have two master files. The files are reversed each new two-week crop calendar period, such that during one two-week period MASTER 1 will be used as input and MASTER 2 as output; the next 14-day period will use MASTER 2 as input and MASTER 1 as output. This procedure is necessary to avoid loss of previously acquired data.

The master used as input contains information on either the last day of the previous crop period or the winter wheat planting date if it has not yet been reached (winter wheat stations only).

If an error is encountered during execution of SUZYQ (ex., inverted or missing temperatures), the mistake must be corrected and SUZYQ resubmitted, using the master files in the same order as the first run.

MASTER 1 is initially created with an IEBGENER IBM utility program, using master cards on file at CCEA/Columbia as the SYSUT1 input.

3.2.1.7 W.EDS.CCEA.{GROUP I / GROUP II}.CROPDATA

RECFM = F
LRECL = 1024
BLKSIZE = 1024
DSORG = PS

This dataset contains the maximum and minimum daily air temperatures (or approximations thereto) for the 14 days for each station listed in INDEX. The CLEAR program is executed to change the starting day of the 14-day period, and also to initialize all values to '9999'. The CAPTURE program fills CROPDATA, and SUZYQ utilizes the file.

3.2.1.8 W.EDS.CCEA.{GROUP I / GROUP II}.GRIDINFO

RECFM = FB
LRECL = 80
BLKSIZE = 12960
DSORG = PS

The GRIDINFO dataset is created or modified as output during execution of SUZYQ. The record contents are in the same format as those of the output master file; while the output master file contains information on only the last day of a two-week period, GRIDINFO contains similar information for every day in the two-week period. GRIDINFO contains only information on stations with successful crop calendar executions; as a result, it is necessary to complete an error-free run of SUZYQ before this dataset is copied to tape using program JSCTAPE.

3.2.1.9 W.EDS.CCEA.{GROUP I / GROUP II}.ARNO

RECFM = VBS
LRECL = 796
BLKSIZE = 800
DSORG = PS

This dataset is created by an IEBPTPCH step after SUZYQ is successfully run, and contains basically the same information as the output master file. It is accessed by JSC via a dial-up telephone line using the IBM Time Sharing Operation (TSO).

3.2.2 SUBROUTINE INIT

INIT is called by CLEAR, CAPTURE, PRINT, EDITOR, INSDEL, and CROPCALN; INIT prepares the weather data file, CROPDATA, for direct access and initializes several variables used in SWAP.

3.2.2.1 Linkages

INIT defines COMMON blocks TALK, INDEX, DATA, and CLOSE, which are used by other subroutines and programs.

3.2.2.2 Interfaces

INIT must be called before SWAP.

3.2.2.3 Inputs

INIT assumes that Unit 2 is attached to the INDEX file and that Unit 4 is attached to file CROPDATA.

3.2.2.4 Outputs

After ejecting one page on the printer, INIT writes a heading for all output produced by subsequent subroutines and main program statements.

3.2.2.5 Description

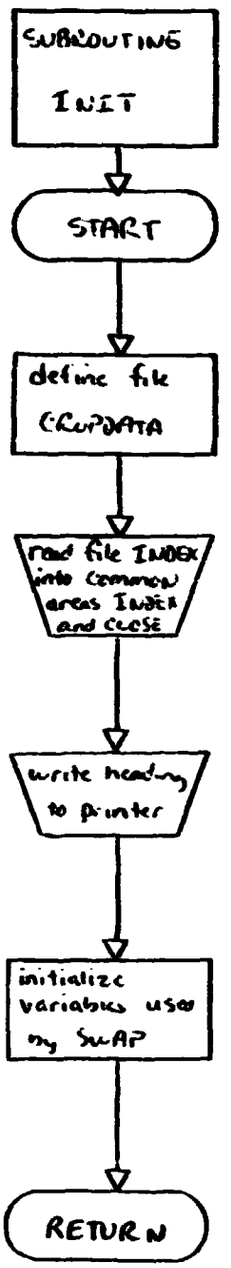
INIT opens file CROPDATA for direct access, reads part of the INDEX file into the COMMON block INDEX and part into COMMON block CLOSE, prints a heading, creates COMMON block DATA, and initializes variables in COMMON block TALK, which is used in SWAP.

3.2.2.6 Flowchart

Next page.

3.2.2.7 Listing

Follows flowchart.


SUBROUTINE
INIT
START
define file
GROUPDATA
read file INDEX
into common
areas INDEX
and CLOSE
write heading
to printer
initialize
variables used
by SWAP
RETURN

RUN NO. 73 DATE 05/04/77 TIME 1820 LISTING OF MODULE INIT

DESCRIPTION INITS AUTOMATION FILE ACCESS

MASTER FILE W,EDC.CCEA.GVH.MASTER
ADDED TO MASTER 07/25/76
LAST DATE COPIED NONE
LAST UPDATE 07/25/76 1809

PASSWORD DGLJ
PROGRAMMER VONHOLT
LANGUAGE FOR
PROC PARAMETER $NO JCL

```
      SUBROUTINE INIT                                                    00000010
      REAL*8 BUFFER(128),BUF2(128)                                       00000020
      INTEGER WMO(511),STATN(511,3)                                      00000030
      INTEGER ACCESS,CHANGE,PAGEI,PAGEO                                  00000040
      INTEGER*2 NUMSTA,FILLER,CCDATA(2,16,16)                            00000050
      COMMON /TALK/INCORE,ACCESS,CHANGE,PAGEI,PAGEO                      00000060
      COMMON /INDEX/NUMSTA,FILLER,WMO                                    00000070
      COMMON /DATA/CCDATA                                                00000080
      COMMON /CLOSE/STATN                                                00000090
      EQUIVALENCE (CCDATA(1,1,1),BUFFER(1))                              00000100
      EQUIVALENCE(NUMSTA,BUF2(1))                                        00000110
C                                                                        00000120
      DEFINE FILE 4(200,1024,L,IDUMMY)                                   00000130   07/25/76
      READ(2,1001)NUMSTA,(WMO(I),(STATN(I,J),J=1,3),I=1,511)             00000140
 1001 FORMAT(A2,2X,255A4,/,10(128A4,/))                                  00000150
      REWIND 2                                                           00000160
      WRITE(6,15)                                                        00000170
   15 FORMAT('1** CCEA PGM CC76-COU-02-02  CROP CALENDAR DATA CAPTURE    00000180   07/25/76
     $    VERSION OF JULY, 1976 **',/,'-')                              00000190   07/25/76
      INCORE = -1                                                        00000200
      ACCESS=0                                                           00000210
      CHANGE=0                                                           00000220
      PAGEI = 0                                                          00000230
      PAGEO=0                                                            00000240
C                                                                        00000250
C                                                                        00000260
      RETURN                                                             00000270
      END                                                                00000280
```

3.2.3 SUBROUTINE SWAP

SWAP is the entry point used by all crop calendar preparation routines to read from or write to file CROPDATA. The 'in' version of SWAP reads from CROPDATA, and the 'out' version writes to CROPDATA.

3.2.3.1 Linkages

SWAP uses the COMMON blocks TALK, INDEX, and DATA, which were declared by subroutine INIT.

3.2.3.2 Interfaces

INIT must be called before SWAP.

3.2.3.3 Inputs and Outputs

INIT assumes Unit 4 is attached to file CROPDATA.

ACTION	if 1, then read from CROPDATA, with TEMPS and JULIAN as outputs to the main program; if 2, then write to CROPDATA, with TEMPS and JULIAN as inputs from the main program.
WMONUM	station number of requested station.
TEMPS	array of 14 days' maximum and minimum temperatures associated with WMONUM.
JULIAN	first date of the 14-day period.
RCODE	if -1, then ACTION was other than 1 or 2; if 0, then transaction was successful; if 10, then WMONUM was not in the INDEX file.

3.2.3.4 Description

SWAP maintains a BUFFER in COMMON block DATA to contain a 16-station block of file CROPDATA; it also maintains a list of crop calendar stations in COMMON INDEX. The list of WMO numbers is in the same order as the data on CROPDATA. To access a WMO station, SWAP sequentially searches INDEX for the requested number. If the number is found, its position in INDEX is used to determine the block address (i.e., in which 16-station block of CROPDATA the station is located). If the station number is not found, RCODE is set to 10. At this point it is determined whether the requested block is in core. If the block is in core, it is read from or written

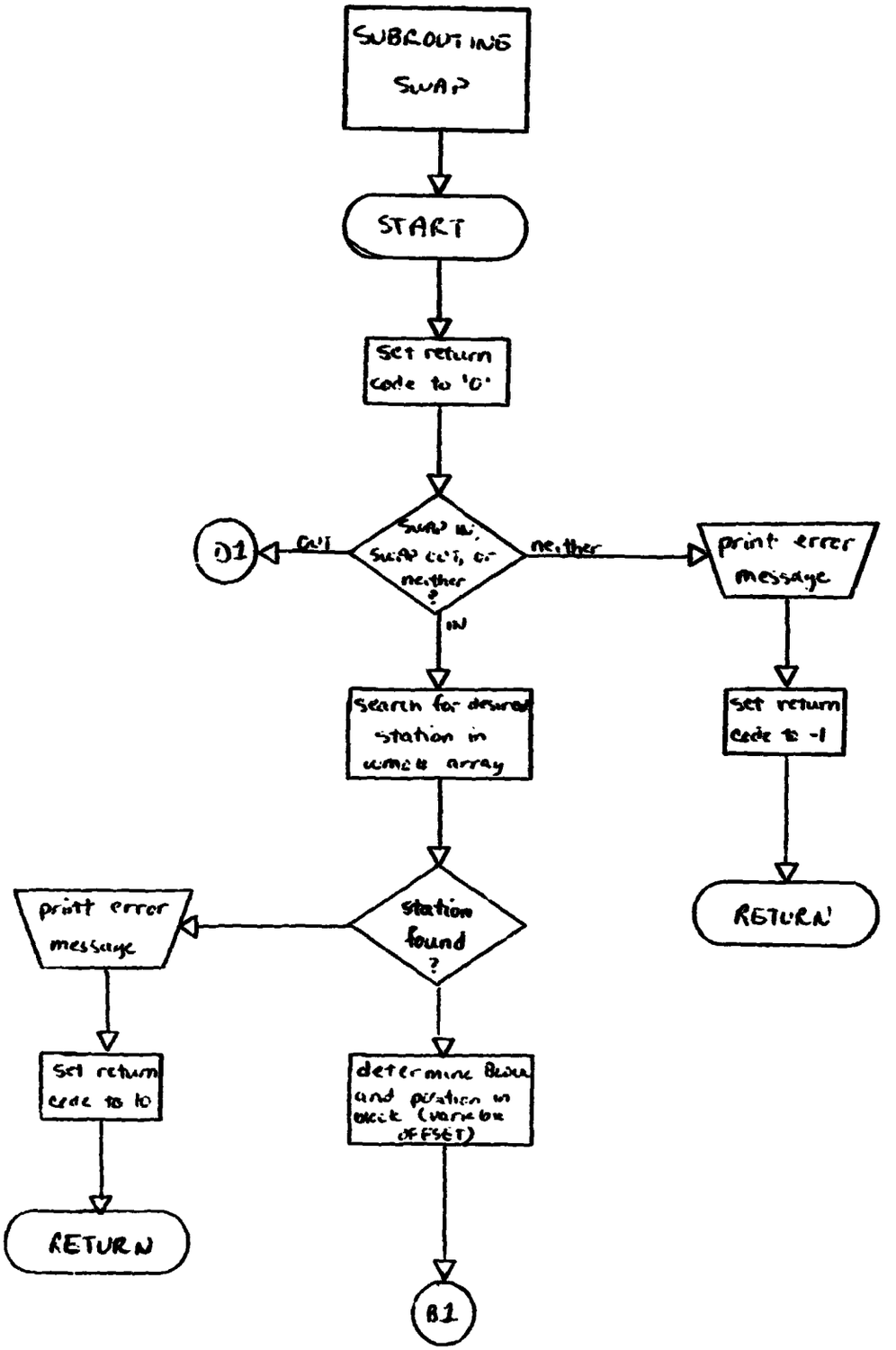
to; if the necessary block is not in core, it is swapped with the block currently in core.

3.2.3.5 Flowchart

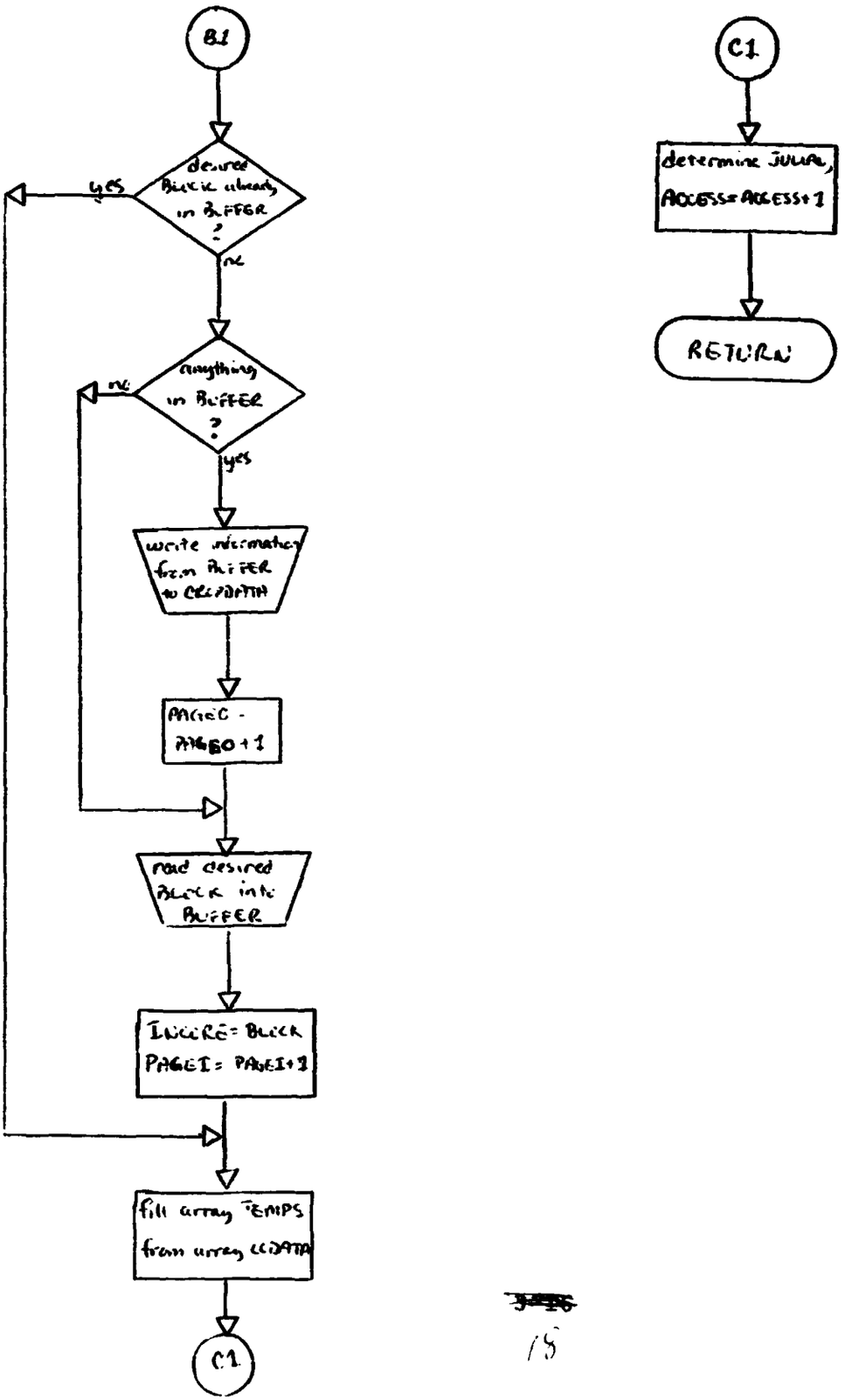
Next page.

3.2.3.6 Listing

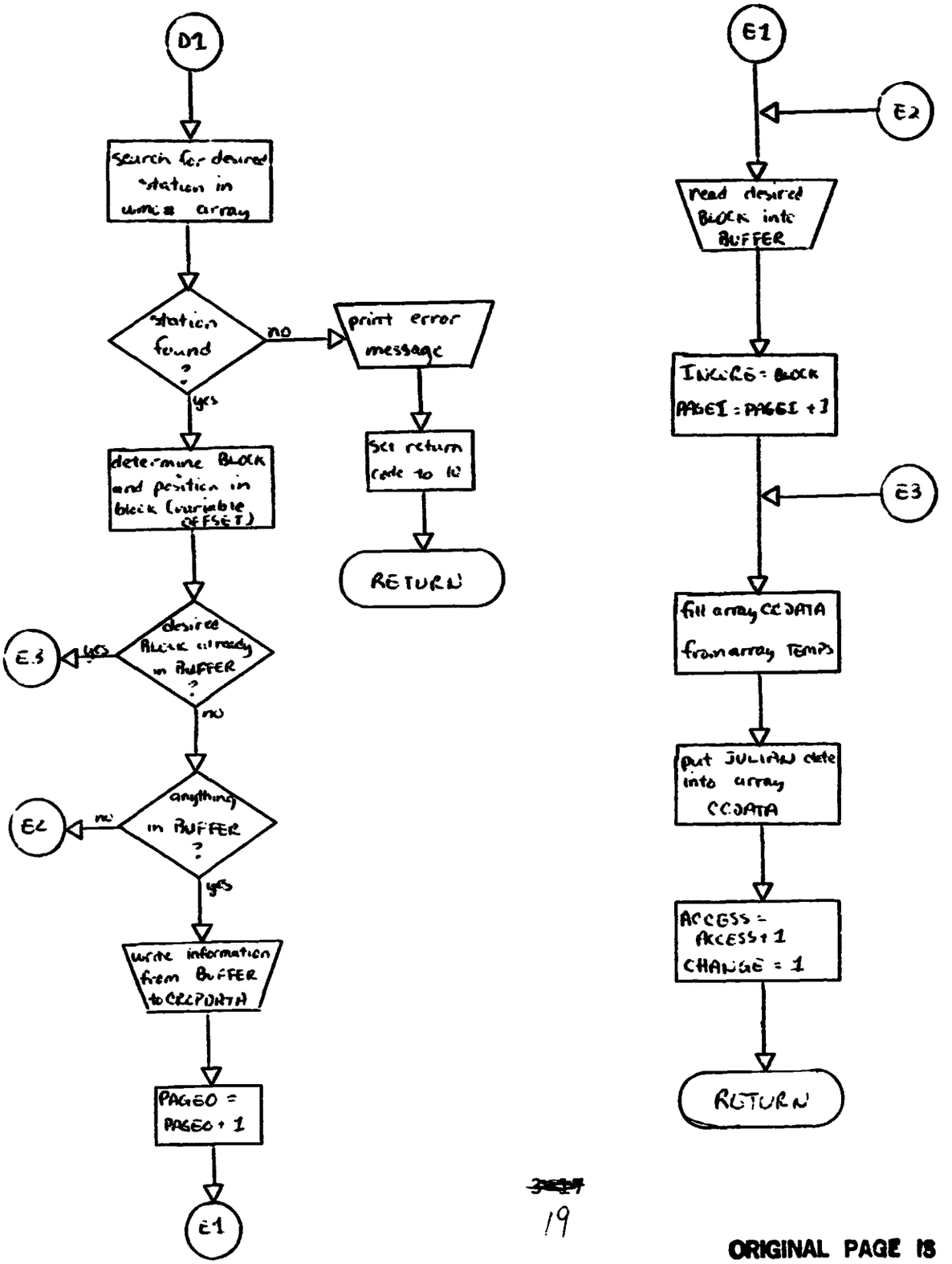
Follows flowchart.


SUBROUTINE SWAP
START
Set return code to '0'
SWAP in, SWAP out, or neither ?
out
B1
neither
print error message
Set return code to -1
RETURN
in
Search for desired station in names array
station found ?
print error message
Set return code to 10
RETURN
determine Block and position in block (variable OFFSET)
B1

SUBROUTINE SWAP, CONT



SUBROUTINE SWAP, CCNIT



RUN NO. 73 DATE 05/04/77 TIME 1820 LISTING OF MODULE SWAP

DESCRIPTION SWAPS MET DATA FOR AUTOMATION

MASTER FILE W.EDS.CCEA.GVM.MASTER
ADDED TO MASTER 07/25/76
LAST DATE COPIED NONE
LAST UPDATE 07/25/76 1719

PASSWORD PDXS
PROGRAMMER VONHOLT
LANGUAGE FOR
PROC PARAMETER $NOJCL

```
      SUBROUTINE SWAP(ACTION,WMONUM,TEMPS, JULIAN,IER)                  00000010   07/25/76
      REAL*8 BUFFER(128)                                                00000020
      INTEGER ACTION,WMONUM,TEMPS(14,2),WMO(511),POSIT                  00000030
      INTEGER ACCESS,CHANGE,PAGEI,PAGEO                                 00000040
      INTEGER BLOCK,OFFSET                                              00000050
      INTEGER*2 NUMSTA,FILLER,CCDATA(2,16,16)                           00000060
      COMMON /TALK/INCORE,ACCESS,CHANGE,PAGEI,PAGEO                     00000070
      COMMON /INDEX/NUMSTA,FILLER,WMO                                   00000080
      COMMON /DATA/CCDATA                                               00000090
      EQUIVALENCE(CCDATA(1,1,1),BUFFER(1))                              00000100
C                                                                       00000110
      IER = 0                                                           00000120   07/25/76
      IF ((ACTION.LT.1).OR.(ACTION.GT.2)) GO TO 3000                    00000130
      GO TO (1000,2000), ACTION                                         00000140
C***  SWAP IN                                                           00000150
 1000 DO 800 J = 1 , NUMSTA                                             00000160
      IF (WMO(J) .EQ. WMONUM)  GO TO 900                                00000170
  800 CONTINUE                                                          00000180
      GO TO 4000                                                        00000190
  900 POSIT = J                                                         00000200
      BLOCK = (POSIT-1) / 16 + 1                                        00000210
      OFFSET=POSIT-(BLOCK-1)*16                                         00000220
      IF (INCORE .EQ. BLOCK)  GO TO 1050                                00000230
      IF (CHANGE) 1040 , 1040 , 1030                                    00000240
 1030 WRITE(4'INCORE,1001) BUFFER                                       00000250
 1001 FORMAT(128A8)                                                     00000260
      PAGEO=PAGEO+1                                                     00000270
 1040 READ(4'BLOCK,1001)BUFFER                                          00000280
      INCORE=BLOCK                                                      00000290
      PAGEI=PAGEI+1                                                     00000300
C                                                                       00000310
 1050 DO 1060 J = 1 , 14                                                00000320
      TEMPS(J,1)=CCDATA(1,J+2,OFFSET)                                   00000330
 1060 TEMPS(J,2)=CCDATA(2,J+2,OFFSET)                                   00000340
      JULIAN=CCDATA(1,1,OFFSET)                                         00000350
      ACCESS=ACCESS+1                                                   00000360
      RETURN                                                            00000370
C***  SWAP OUT                                                          00000380
 2000 DO 1800 J = 1 , NUMSTA                                            00000390
      IF (WMO(J) .EQ. WMONUM)  GO TO 1900                               00000400
 1800 CONTINUE                                                          00000410
      GO TO 4000                                                        00000420
 1900 POSIT=J                                                           00000430
```

```
      BLOCK = (POSIT-1) / 16 + 1                                        00000440
      OFFSET=POSIT-(BLOCK-1)*16                                         00000450
      IF (INCORE .EQ. BLOCK)  GO TO 2050                                00000460
      IF (CHANGE) 2040, 2040 , 2030                                     00000470
 2030 WRITE(4'INCORE,1001)BUFFER                                        00000480
      PAGEO=PAGEO+1                                                     00000490
 2040 READ(4'BLOCK,1001)BUFFER                                          00000500
      INCORE=BLOCK                                                      00000510
      PAGEI=PAGEI+1                                                     00000520
C                                                                       00000530
 2050 DO 2060 J = 1 , 14                                                00000540
      CCDATA(1,J+2,OFFSET)=TEMPS(J,1)                                   00000550
 2060 CCDATA(2,J+2,OFFSET)=TEMPS(J,2)                                   00000560
      CCDATA(1,1,OFFSET)=JULIAN                                         00000570
      ACCESS=ACCESS+1                                                   00000580
      CHANGE=1                                                          00000590
      RETURN                                                            00000600
C                                                                       00000610
 3000 WRITE(6,3010)ACTION                                               00000620
 3010 FORMAT('0INVALID ACTION CODE:',I7)                                00000630
      CALL ERRTRA                                                       00000640
      STOP                                                              00000650
C                                                                       00000660
 4000 IER = 1                                                           00000670
      RETURN                                                            00000680
C                                                                       00000690
      END                                                               00000700
```

07/25/76
07/25/76

3.2.4 SUBROUTINE TERM

TERM is used by all crop calendar preparation routines to close the weather file and print statistics accumulated during the running of SWAP.

3.2.4.1 Linkages

TERM utilizes COMMON blocks TALK and DATA.

3.2.4.2 Interfaces

SWAP must be called before TERM. TERM is the last subroutine called before the end of any crop calendar preparation program.

3.2.4.3 Inputs

TERM uses COMMON blocks TALK and DATA.

3.2.4.4 Outputs

TERM may write a block to CROPDATA, assumed on Unit 4. The subroutine also prints out SWAP 'IN'/SWAP 'OUT' statistics and a closing message.

3.2.4.5 Description

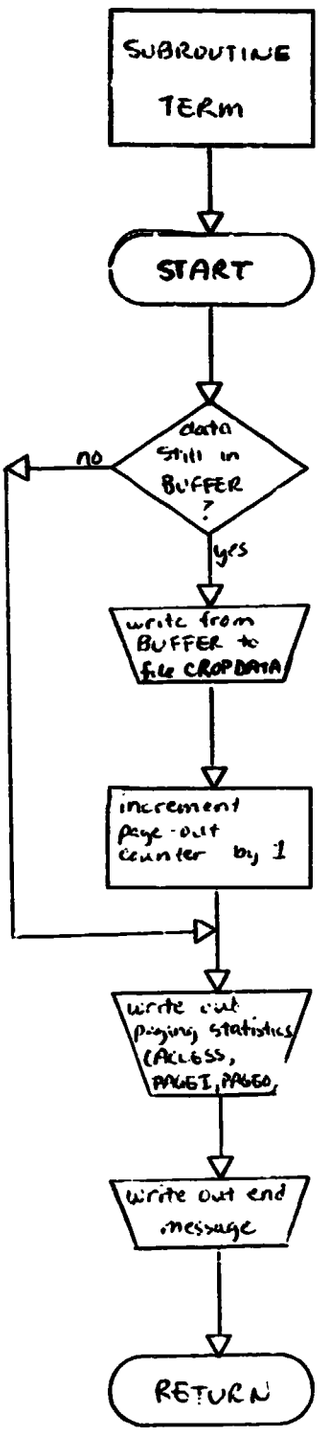
If a 16-station block remains in BUFFER, TERM writes the block to CROPDATA. Statistics accumulated during the execution of SWAP and a closing message are written to the printer.

3.2.4.6 Flowchart

Next page.

3.2.4.7 Listing

Follows flowchart.


SUBROUTINE
TERM
START
data
still in
BUFFER
?
no
yes
write from
BUFFER to
file CROPDATA
increment
page-out
counter by 1
write out
paging statistics
(ACCESS,
PAGEI, PAGEO)
write out end
message
RETURN

RUN NO. 73 DATE 05/04/77 TIME 1820 LISTING OF MODULE TERM

DESCRIPTION TERM AUTOMATION FILE

MASTER FILE W.EDS.CCEA.GVH.MASTER
ADDED TO MASTER 07/25/76
LAST DATE COPIED NONE
LAST UPDATE NONE

PASSWORD HLSK
PROGRAMMER VONHOLT
LANGUAGE FOR
PROC PARAMETER $NOJCL

```
      SUBROUTINE TERM                                                    00000010
      REAL*8 BUFFER(128)                                                 00000020
      INTEGER ACCESS,CHANGE,PAGEI,PAGEO                                  00000030
      COMMON /TALK/INCORE,ACCESS,CHANGE,PAGEI,PAGEO                      00000040
      COMMON /DATA/BUFFER                                                00000050
C                                                                        00000060
      IF (CHANGE) 200 , 200 , 100                                        00000070
  100 WRITE(4'INCORE,110)BUFFER                                          00000080
  110 FORMAT(128A8)                                                      00000090
      PAGEO=PAGEO+1                                                      00000100
  200 WRITE(6,210)ACCESS,PAGEI,PAGEO                                     00000110
  210 FORMAT( '-** PAGING STATISTICS **',/,'-ACCESSES',I15,/,            00000120
     *   '-PAGE INS',I15,/,'-PAGE OUTS',I14)                             00000130
      WRITE(6,1)                                                         00000140
    1 FORMAT('-',/,'-',/,'-** END OF CAPTURE ROUTINE EXECUTION **')      00000150
      RETURN                                                             00000160
      END                                                                00000170
```

3.2.5 PROGRAM CLEAR

CLEAR resets all entries in CROPDATA to missing values (9999) and calculates the starting Julian date for the 14-day period about to be processed.

3.2.5.1 Linkages

CLEAR calls subroutines INIT, SWAP, AND TERM and uses COMMON block INDEX.

3.2.5.2 Interfaces

CLEAR is the first crop calendar preparation routine to be run for any given 14-day period.

3.2.5.3 Inputs

The INDEX file is on Unit 2, CROPDATA is on Unit 4, and optional card input (date - reset value) on Unit 5.

3.2.5.4 Outputs

The CROPDATA file is initialized.

3.2.5.5 Description

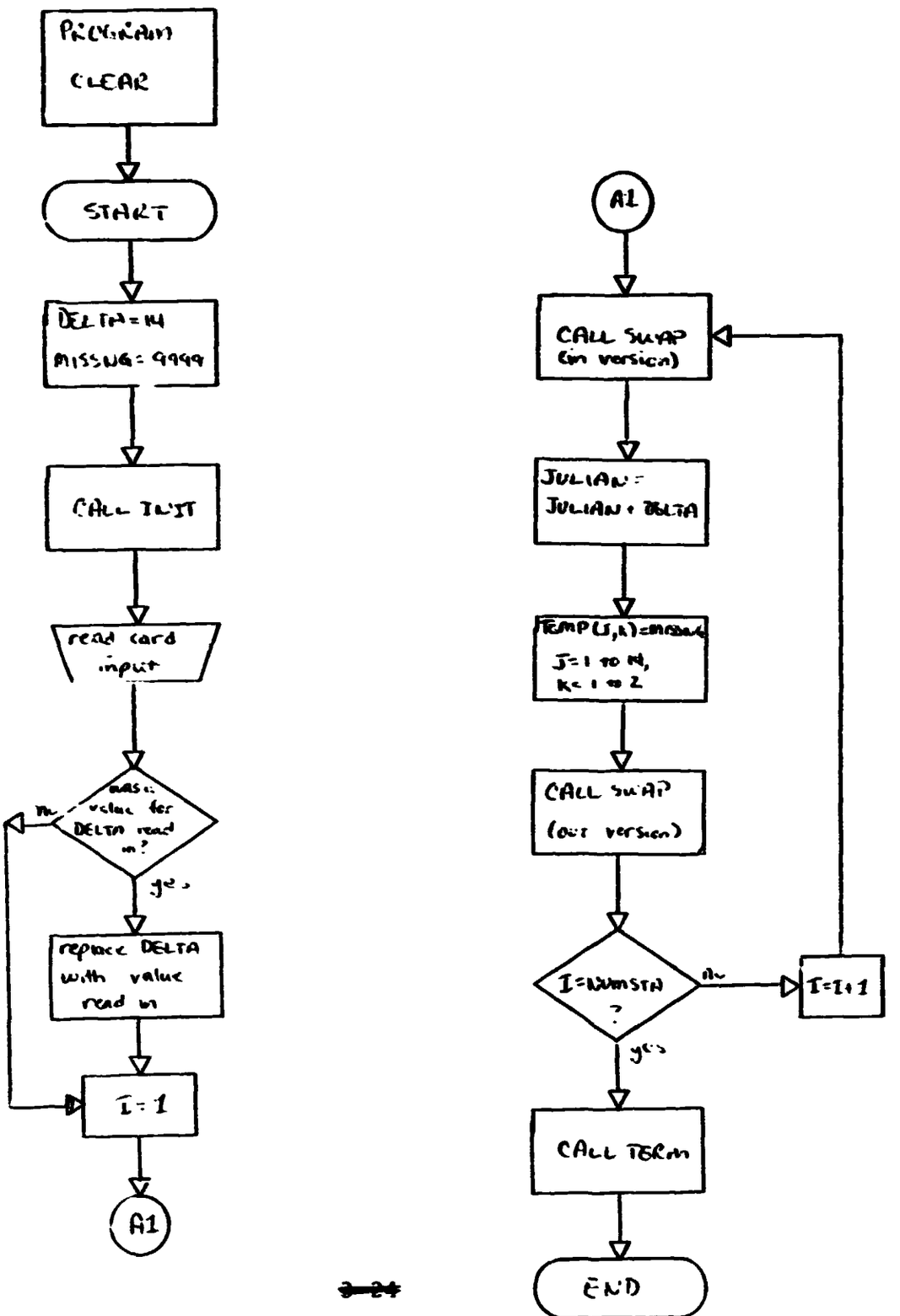
CLEAR calls INIT to load COMMON block INDEX and open CROP-DATA to direct access. An attempt is made to read a date-reset value from Unit 5; if no value is present, the reset value defaults to 14 days. The starting date of the 14-day period to be processed is reset and all temperatures in CROPDATA are set to '9999', using SWAP. The file is then closed by subroutine TERM.

3.2.5.6 Flowchart

Next page.

3.2.5.7 Listing

Follows flowchart.


PROGRAM
CLEAR
START
DELTA = 14
MISSING = 9999
CALL INIT
read card input
missing value for DELTA read in?
no
yes
replace DELTA with value read in
I = 1
A1
A1
CALL SWAP (in version)
JULIAN = JULIAN + DELTA
TEMP(I,K)=MASTER
J = 1 TO N,
K = 1 TO 2
CALL SWAP (out version)
I=NUMSTA ?
no
I=I+1
yes
CALL TERM
END

RUN NO. 73 DATE 05/04/77 TIME 1820 LISTING OF MODULE CLEAR

DESCRIPTION CLEARS AUTOMATION FILE

MASTER FILE W,EDS,CCEA,GVH,MASTER
ADDED TO MASTER 07/25/76
LAST DATE COPIED NONE
LAST UPDATE 07/25/76 2129

PASSWORD SLFH
PROGRAMMER VONHOLT
LANGUAGE FOR
PROC PARAMETER $NO,JCL

```
      INTEGER WMO(511),TEMPS(14,2),JULIAN,IER,IN/1/,OUT/2/,MISSNG/9999/   00000010
      INTEGER DELTA/14/                                                   00000020   07/25/76
      INTEGER*2 NUMSTA,FILLER                                             00000030
      COMMON /INDEX/NUMSTA,FILLER,WMO                                     00000040
C                                                                         00000050
      CALL INIT                                                           00000060
      READ(5,9,END=2)DELTA                                                00000070   07/25/76
    2 CONTINUE                                                            00000080   07/25/76
      DO 10 I = 1, NUMSTA                                                 00000090
      CALL SWAP(IN,WMO(I),TEMPS,JULIAN,IER)                               00000100
      JULIAN = JULIAN + DELTA                                             00000110   07/25/76
      DO 5 J = 1, 14                                                      00000120
      DO 5 K = 1, 2                                                       00000130
    5 TEMPS(J,K) = MISSNG                                                 00000140
      CALL SWAP(OUT,WMO(I),TEMPS,JULIAN,IER)                              00000150
   10 CONTINUE                                                            00000160
      CALL TERM                                                           00000170
C                                                                         00000180
      STOP                                                                00000190
      END                                                                 00000200
```

3.2.6 JCL FILE CLEAR

Executing JCL file CLEAR invokes the CLEAR program.

3.2.6.1 Linkages

None.

3.2.6.2 Interfaces

None.

3.2.6.3 Inputs

See Sections 3.2.5.3 and 4.2.2.1.

3.2.6.4 Outputs

None.

3.2.6.5 Description

JCL file CLEAR binds the files referenced by program CLEAR to that program and requests that the program be executed.

3.2.6.6 Listing

```
//CLEARØPROC
//CLEARØEXECØPGM=CLEAR
//STEPLIBØDDØDSN=W.EDS.CCEA.PHASE3.LOAD.DISP=SHR
//FT02F001ØDDØDSN=W.EDS.CCEA.{GROUPI }.INDEX,DISP=SHR
                            {GROUPII}
//FT04F001ØDDØDSN=W.EDS.CCEA.{GROUPI }.CROPDATA,DISP=SHR
                            {GROUPII}
//FT06F001ØDDØSYSOUT=A
//FT05F001ØDDØDDNAME=SYSIN
//ØØØØØØØPEND
```

3.2.7 PROGRAM CAPTURE

CAPTURE reads the tape and disk temperature data files built at CCEA/Washington and loads them into CROPDATA.

3.2.7.1 Linkages

CAPTURE calls subroutines INIT, CVB, SWAP, and TERM.

3.2.7.2 Interfaces

CLEAR must be run before CAPTURE.

3.2.7.3 Inputs

CAPTURE assumes the INDEX file is on Unit 2, the tape or disk data file on Unit 3, CROPDATA on Unit 4, and an optional card input file on Unit 5. This latter file is used with North American data only; it contains the Julian date of the first day of the month being processed, and the new year if the 2-week period spans two years.

3.2.7.4 Outputs

CAPTURE loads CROPDATA with temperature data for the 14-day period being processed.

3.2.7.5 Description

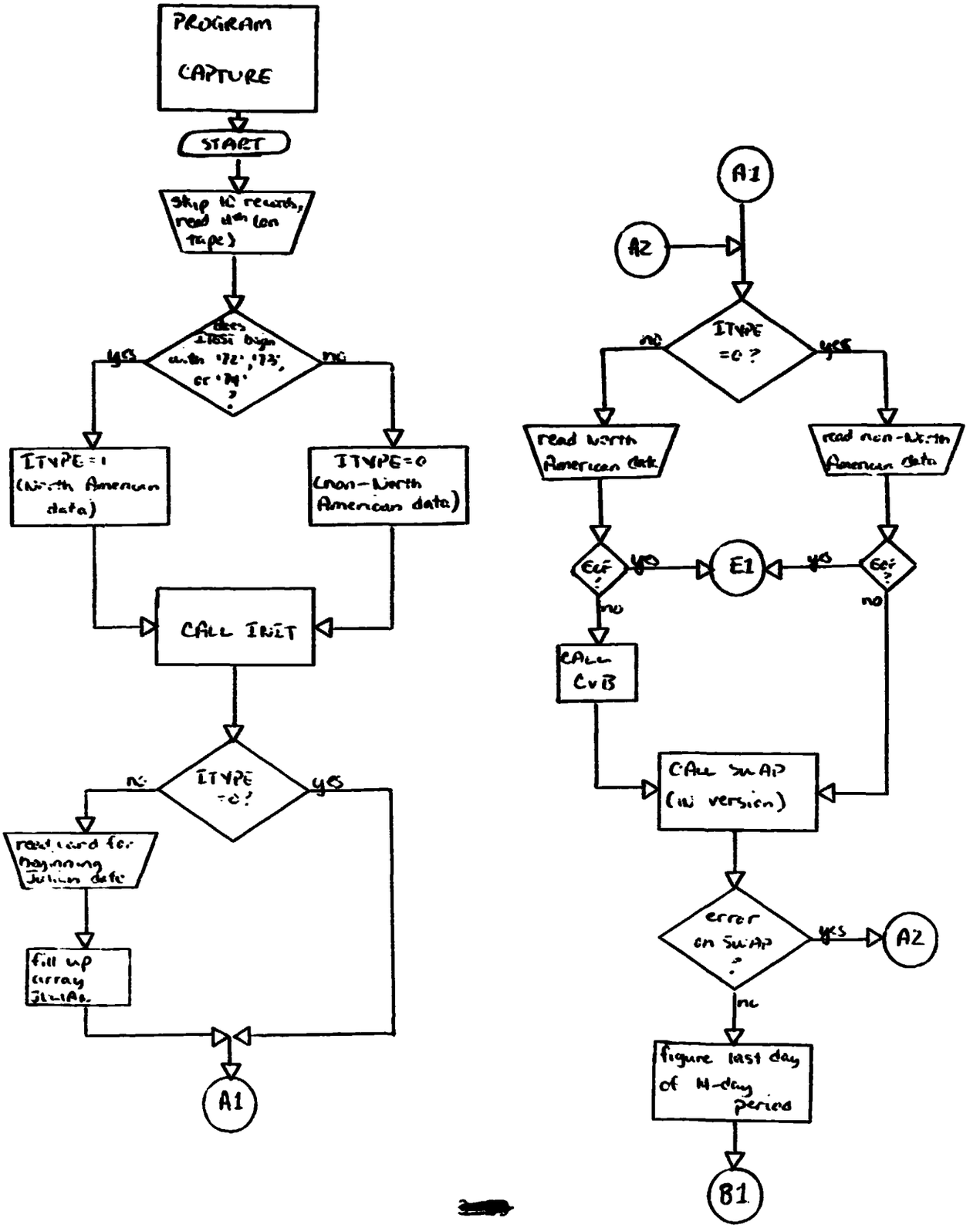
CAPTURE reads the eleventh record from Unit 3 to determine if the data is "North American" or "non-North American". Unit 3 is rewound and INIT is called. For North American data, the Julian date of the first day of the month is read from Unit 5 and the array JULIAN is filled. North American data also requires that CAPTURE call subroutine CVB. For each new station SWAP(in) is called; CAPTURE determines what portion of the 14-day crop calendar is being accessed by this run of the program. CAPTURE makes the necessary adjustments if the 2-week period spans two years, including reading another card from Unit 5 for North American data to obtain the new year. Using SWAP(out), CAPTURE updates the CROPDATA file with the new weather data. After all stations have been processed, TERM is called and processing ceases.

3.2.7.6 Flowchart

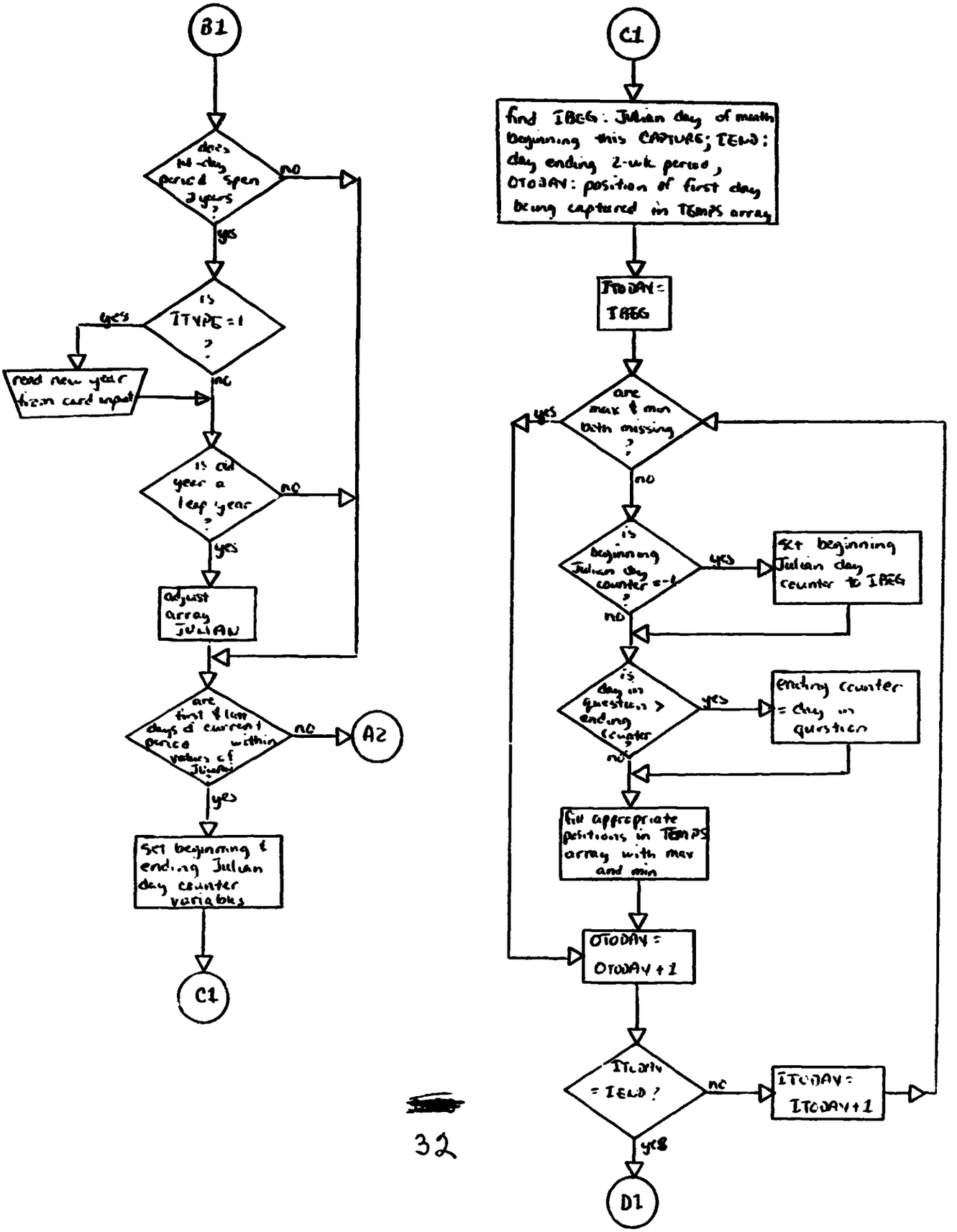
Next page.

3.2.7.7 Listing

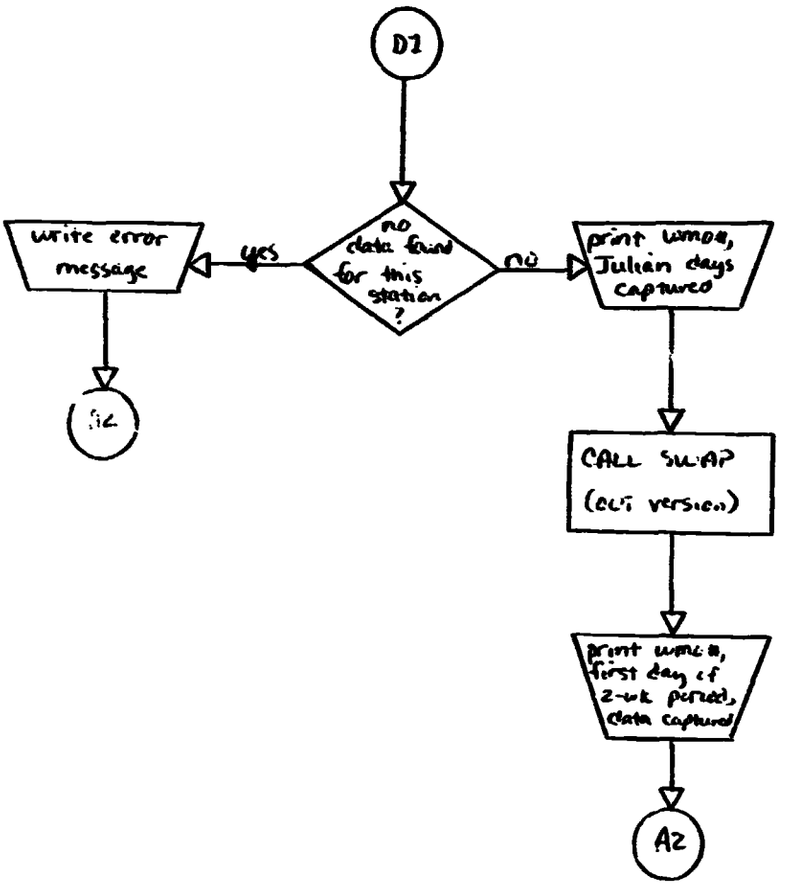
Follows flowchart.



PROGRAM CAPTURE, CON'T



PROGRAM CAPTURE, CONT



RUN NO. 70 DATE 04/28/77 TIME 1633 LISTING OF MODULE CAPTURE

DESCRIPTION AUTOMATION [redacted] CAPTURE RTN.

MASTER FILE A.FDS.CCFA.GVH.MASTER
ADDED TO MASTER 07/25/76
LAST DATE COPIED NONE
LAST UPDATE 01/10/77 1632

PASSWORD KLHS
PROGRAMMER VONHOLT
LANGUAGE FOR
PROC PARAMETER SUB.JCL


```
      INTEGER A7200/7200/,A7499/7499/,WMO(2)                                00000010   07/25/76
      INTEGER HIGH(31),LOW(31),TEMPS(14,2),IN/1/,OUT/2/                     00000020
      INTEGER I,WMONM,IER,JDAY,LDAY,OTODAY,IBEG,IEND,ITODAY,VBEG,VEND       00000030
      INTEGER*2 HI(31),LO(31),JULIAN(31),IYR                                00000040   01/10/77
C                                                                           00000050
      READ(3,1)ITEST                                                        00000060   07/25/76
    1 FORMAT(//////////,4X,A4)                                              00000070   12/15/76
      ITYPE = 0                                                             00000080   07/25/76
      IF ((ITEST.GE.A7200).AND.(ITEST.LE.A7499)) ITYPE = 1                  00000090   07/25/76
      REWIND 3                                                              00000100   07/25/76
      CALL INIT                                                             00000110
C                                                                           00000120
      IF (ITYPE.EQ.0)  GO TO 25                                             00000130   07/25/76
      READ(5,11)JULIAN(1)                                                   00000140
   11 FORMAT(I3)                                                            00000150
      DO 13 I = 2 , 31                                                      00000160
   13 JULIAN(I) = JULIAN(I-1) + 1                                           00000170
C                                                                           00000180
C                                                                           00000190
   20 IF (ITYPE.EQ.0)  GO TO 25                                             00000200   07/25/76
      READ(3,21,END=100)WMO,(LOW(I),HIGH(I),I=1,31)                         00000210   10/18/76
   21 FORMAT(4X,A4,A1,11X,31(2A4,4X))                                       00000220
      IF ((WMO(1).LT.A7200).OR.(WMO(1).GT.A7499))  GO TO 20                 00000230
      CALL CVB(WMO,2,WMONM)                                                 00000240
      GO TO 29                                                              00000250   07/25/76
   25 READ(3,26,END=100)WMONM,IYR,(LO(I),HI(I),JULIAN(I),I=1,31)            00000260   01/10/77
   26 FORMAT(4X,A4,A2,31(2A2,4X,A2,2X))                                     00000270   01/10/77
      DO 27 I = 1 , 31                                                      00000280   07/25/76
      LOW(I) = LO(I)                                                        00000290   07/25/76
   27 HIGH(I) = HI(I)                                                       00000300   07/25/76
   29 CALL SWAP(IN,WMONM,TEMPS,JDAY,IER)                                    00000310   07/25/76
      IF (IER) 20 , 30 , 20                                                 00000320
   30 LDAY = JDAY + 13                                                      00000330
C                                                                           00000340   01/10/77
C CHECK IF PERIOD SPANS TWO YEARS AND IF DATA IS FROM BEGINNING OF          00000350   01/10/77
C  THE SECOND YEAR                                                          00000360   01/10/77
C THEN CHECK IF DATA IS FROM NORTH AMERICA. IF SO, READ IN SECOND YEAR       00000370   01/10/77
C                                                                           00000380   01/10/77
      J=0                                                                   00000390   01/10/77
      IF(LDAY.LE.365..OR.JULIAN(1).GE.335)GO TO 35                          00000400   01/10/77
      IF(ITYPE.EQ.1)READ(5,32)IYR                                           00000410   01/10/77
   32 FORMAT(I4)                                                            00000420   01/10/77
      YR=IYR                                                                00000430   01/10/77
```

```
      C                                                                 00000440  01/10/77
      C CHECK FOR LEAP YEAR                                             00000450  01/10/77
      C                                                                 00000460  01/10/77
            CHECK=(YR-1.0)/4                                            00000470  01/10/77
            ICHECK=CHECK                                                00000480  01/10/77
            IF(CHECK.EQ.ICHECK)J=1                                      00000490  01/10/77
            IF(J.EQ.1 .AND. LDAY.LE.366)GO TO 35                        00000500  01/10/77
      C                                                                 00000510  01/10/77
      C ADJUST JULIAN DATE                                              00000520  01/10/77
      C                                                                 00000530  01/10/77
            DO 36 I=1,3                                                 00000540  01/10/77
            JULIAN(I)=JULIAN(I)+365+J                                   00000550  01/10/77
         36 CONTINUE                                                    00000560  01/10/77
         35 CONTINUE                                                    00000570  01/10/77
            IF ((LDAY.LT.JULIAN(1)).OR.(JDAY.GT.JULIAN(1)+30))  GO TO 20 00000580  07/25/76
            VBEG = -1                                                   00000590
            VEND = -1                                                   00000600  07/25/76
            IF (JDAY.LE.JULIAN(1))  GO TO 40                            00000610
            IBEG = JDAY + 1 - JULIAN(1)                                 00000620
            OTODAY = 1                                                  00000630
            IEND = MIND(31,IBEG+13)                                     00000640  07/25/76
            GO TO 50                                                    00000650
         40 IBEG = 1                                                    00000660
            OTODAY = JULIAN(1) + 1 - JDAY                               00000670
            IEND = 15 - OTODAY                                          00000680
         50 DO 60 ITODAY = IBEG , IEND                                  00000690
            IF ((LOW(ITODAY).EQ.9999).AND.(HIGH(ITODAY).EQ.9999))  GO TO 60 00000700
            IF (VBEG.EQ.-1)VBEG = JULIAN(ITODAY)                        00000710
            IF (JULIAN(ITODAY).GT.VEND)VEND=JULIAN(ITODAY)              00000720  07/25/76
            TEMPS(OTODAY,1) = LOW(ITODAY)                               00000730
            TEMPS(OTODAY,2) = HIGH(ITODAY)                              00000740
         60 OTODAY = OTODAY + 1                                         00000750
            IF (VBEG.EQ.-1)  GO TO 65                                   00000760
            WRITE(6,63)WMONM,VBEG,VEND                                  00000770  07/25/76
         63 FORMAT(1X,15,'(',I3,'-',I3,')')                             00000780  07/25/76
            CALL SWAP(OUT,WMONM,TEMPS,JDAY,IER)                         00000790
            WRITE(1,64)WMONM,JDAY,(TEMPS(J,2),J=1,14),WMONM,JDAY,       00000800  07/25/76
           $   (TEMPS(J,1),J=1,14)                                      00000810  07/25/76
         64 FORMAT(15,' MX',14,' 14',1416,/,15,' MN',14,' 14',1416)     00000820  07/25/76
            GO TO 20                                                    00000830
         65 WRITE(6,67)WMONM                                            00000840  07/25/76
         67 FORMAT(16,'( NONE )')                                       00000850  07/25/76
            GO TO 20                                                    00000860
        100 CALL TERM                                                   00000870
            STOP                                                        00000880
            END                                                         00000890
```

3.2.8 JCL FILE CAPTURE

Executing JCL file CAPTURE involves the CAPTURE program.

3.2.8.1 Linkages

None.

3.2.8.2 Interfaces

None.

3.2.8.3 Inputs

Direct card inputs to this file are necessary only with North American data. See Sections 3.2.7.3 and 4.2.2.2.

3.2.8.4 Outputs

None.

3.2.8.5 Description

JCL file CAPTURE binds the files referenced by program CAPTURE to that program and requests invocation of program CAPTURE.

3.2.8.6 Listing

```
//CAPTURE ØPROC ØUNIT=,VOL=,DSN='W.EDS.CCEA.DAYDATA'
//CAPTURE ØEXEC ØPGM=CAPTURE
//STEPLIB ØDD ØDSN=W.EDS.CCEA.PHASE3.LOAD,DISP=SHR
//FT01F001 ØDD ØSYSOUT=A,DCB=(LRECL=100,BLKSIZE=1500,RECFM=FB)
//FT02F001 ØDD ØDSN=W.EDS.CCEA.{GROUPI  }.INDEX,DISP=SHR
                               {GROUPII }
//FT03F001 ØDD ØDSN=&DSN.,DISP=SHR,UNIT=&UNIT.,VOL=SER=&VOL,
//ØØØØ LABEL=(,,,IN)
//FT04F001 ØDD ØDSN=W.EDS.CCEA.{GROUPI  }.CROPDATA.DISP=SHR
                               {GROUPII }
//FT06F001 ØDD ØSYSOUT=A
//ØØØØØ PEND
```

3.2.9 SUBROUTINE CVB

CVB is called by CAPTURE to convert WMO numbers from zoned decimal to binary (North American data only).

3.2.9.1 Linkages

None.

3.2.9.2 Interfaces

None.

3.2.9.3 Inputs and Outputs

FROM	the input WMO number (zoned decimal)
NCHAR	number of bytes in FROM (in this case, NCHAR=5
TO	the output WMO number (binary)

3.2.9.4 Description

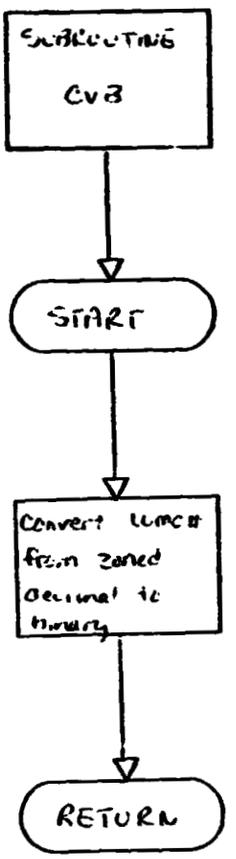
North American WMO numbers exist as zoned decimal and need to be converted to binary before processing can take place. CVB accomplishes this conversion.

3.2.9.5 Flowchart

Next page.

3.2.9.6 Listing

Follows flowchart.


SUBROUTINE
CVB
START
Convert LUNC#
from zoned
decimal to
binary
RETURN

```
//STEP1 EXEC ASMHC
//ASM.SYSIN DD *
CVB       TITLE '- CONVERTS FROM ZONED DECIMAL TO BINARY'
CVB       CSECT
          SPACE
*********************************************************************
*
*         SUBROUTINE CVB(FROM,NCHAR,TO)
*
*
*         PUPPOSE   CONVERT NCHAR CHARACTERS(BYTES) FROM ZONED DECIMAL
*                   (EBCDIC) TO BINARY, STORING FULLWORD RESULT AT
*                   LOCATION TO.    NCHAR MUST BE BETWEEN 1 AND 16,
*                   INCLUSIVE.
*
*         LANGUAGE  ASSEMBLER
*
*         WRITTEN BY G VON HOLT, JULY 1976
*                   FORTRAN X + H COMPATIBLE
*
*********************************************************************
          SPACE 2
          USING *,15                      ADDRESSABILITY
          B     BEGIN                     BRANCH AROUND NAME
          DC    AL1(3)                    LENGTH OF NAME
          DC    CL3'CVB'                  NAME OF SUBROUTINE
BEGIN     STM   14,12,12(13)              SAVE CALLER'S REGISTERS
          LM    3,5,0(1)                  GET ADDRESS OF PARMS.
          L     4,0(4)                    GET NUMBER OF CHARACTERS
          BCTR  4,0                       SUBTRACT 1 FOR EX INST.
          XC    DBL,DBL                   CLEAR RESULT FIELD
          EX    4,PACK                    CONVERT TO PACKED DECIMAL
          CVB   6,DBL                     CONVERT PACKED DEC TO BINARY
          ST    6,0(5)                    AND PLACE IN RESULT AREA
          LM    14,12,12(13)              RESTORE CALLERS REGISTERS
          BR    14                        RETURN
          SPACE 2
*
*         OBJECT OF EX INST.
*
PACK      PACK  DBL+4(4),0(0,3)           CONVERTS TO PACKED DECIMAL
          SPACE 2
*
*         SPACE FOR PACKED DECIMAL NUMBER
*
DBL       DS    D
          SPACE 2
          END
/*
```

3.2.10 PROGRAM PRINT

PRINT lists the contents of the CROPDATA file. It may be run at any time.

3.2.10.1 Linkages

PRINT calls INIT, SWAP, and TERM and uses COMMON block INDEX.

3.2.10.2 Interfaces

None.

3.2.10.3 Inputs

PRINT assumes the INDEX file is on Unit 2 and the CROPDATA file is on Unit 4.

3.2.10.4 Outputs

PRINT lists each of the stations on file INDEX and the corresponding temperature data from file CROPDATA.

3.2.10.5 Description

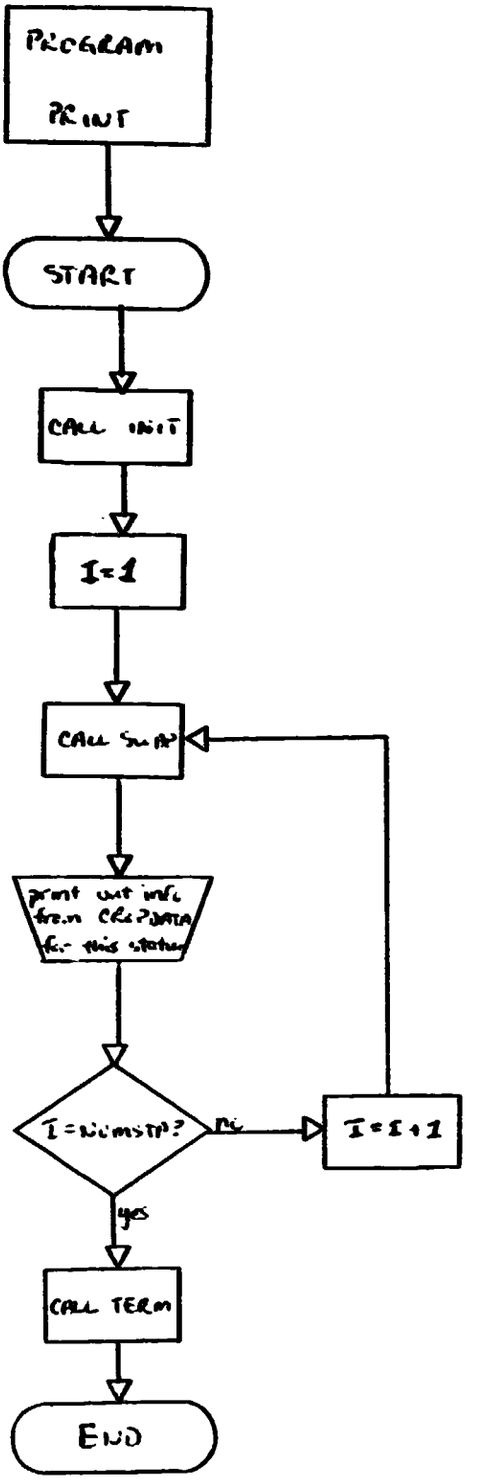
PRINT calls INIT and proceeds to call SWAP for each station on the INDEX file, printing the station number, starting Julian date of the current 2-week period, and the maximum and minimum temperatures in the 2-week period. After information for all stations on INDEX has been listed, TERM is called.

3.2.1.6 Flowchart

Next page.

3.2.10.7 Listing

Follows flowchart.


PROGRAM
PRINT
START
CALL INIT
I=1
CALL SUMP
print unit info
from CREDATA
for this status
I=NUMSTA?
no
I=I+1
yes
CALL TERM
END

RUN NO. 73 DATE 05/04/77 TIME 1820 LISTING OF MODULE PRINT

DESCRIPTION PRINTS OUT AUTOMATION FILE

MASTER FILE W.EDS.CCEA.GVH.MASTER
ADDED TO MASTER 07/25/76
LAST DATE COPIED NONE
LAST UPDATE 07/25/76 2014

PASSWORD HXFH
PROGRAMMER VONHOLT
LANGUAGE FOR
PROC PARAMETER $NOJCL

```
      INTEGER WMO(511),IN/1/,TEMPS(14,2),JULIAN,IER                    00000010
      INTEGER*2 NUMSTA,FILLER                                          00000020
      COMMON /INDEX/NUMSTA,FILLER,WMO                                  00000030
      CALL INIT                                                        00000040
      WRITE (6,5)                                                      00000050
    5 FORMAT('1')                                                      00000060
      DO 10 I = 1 , NUMSTA                                             00000070
      CALL SWAP(IN,WMO(I),TEMPS,JULIAN,IER)                            00000080
   10 WRITE(6,20)WMO(I),JULIAN,(TEMPS(J,2),J=1,14),                    00000090
     $            WMO(I),JULIAN,(TEMPS(J,1),J=1,14)                   00000100
   20 FORMAT(I6,' MAX',I5,2X,14I6,/,                                   00000110
     $       I6,' MIN',I5,2X,14I6,/)                                   00000120
      WRITE(6,5)                                                       00000130
      CALL TERM                                                        00000140
      STOP                                                             00000150
      END                                                              00000160
```

07/25/76

3.2.11 JCL FILE LIST

Executing JCL file LIST invokes the PRINT program.

3.2.11.1 Linkages

See Section 3.2.10.1.

3.2.11.2 Interfaces

None.

3.2.11.3 Inputs

None.

3.2.11.4 Outputs

None.

3.2.11.5 Description

JCL file LIST binds the files referenced by program PRINT to that program and requests execution of program PRINT.

3.2.11.6 Listing

```
//LISTØPROC
//LISTØEXECØPGM=PRINT
//STEPLIBØDDØDSN=W.EDS.CCEA.PHASE3.LOAD,DISP=SHR
//FT02F001ØDDØDSN=W.EDS.CCEA.{GROUPI  }.INDEX,DISP=SHR
                             {GROUPII }
//FT04F001ØDDØDSN=W.EDS.CCEA.{GROUPI  }.CROPDATA,DISP=SHR
                             {GROUPII }
//FT06F001ØDDØSYSOUT=A
//ØØØØØPEND
```

3.2.12 PROGRAM EDITOR

EDITOR checks for inverted temperatures of missing data and attempts to replace bad values by substituting data from a given station's three closest neighbor stations.

3.2.12.1 Linkages

EDITOR calls, INIT, SWAP, CHECK, REPLAC, and TERM and uses COMMON blocks INDEX, VAL1, and CLOSE.

3.2.12.2 Interfaces

EDITOR is not executed until all runs of CAPTURE have been completed and is run before the CROPCALN program is executed.

3.2.12.3 Inputs

EDITOR assumes that the INDEX file is on Unit 2, CROPDATA on Unit 4, and card input on Unit 5, containing the variables IP, IC, IE, and IS.

These four variables attain the following values:

```
IP = 0 no listing
   = 1 list the file CROPDATA
IC = 0 do not call CHECK or REPLAC
   = 1 call CHECK but do not print results
   = 2 call CHECK and print results
IE = 0 do not call REPLAC
   = 1 call REPLAC
IS = 0 do not rewrite CROPDATA
   = 1 rewrite CROPDATA
```

3.2.12.4 Outputs

EDITOR produces corrections for CROPDATA and writes messages to the printer.

3.2.12.5 Description

Program EDITOR calls INIT and, using SWAP, fills arrays IAP and XTEMP. IAP contains the portion of the INDEX file contained in COMMON block CLOSE (the three closest neighbor stations for each station on the INDEX file); XTEMP contains

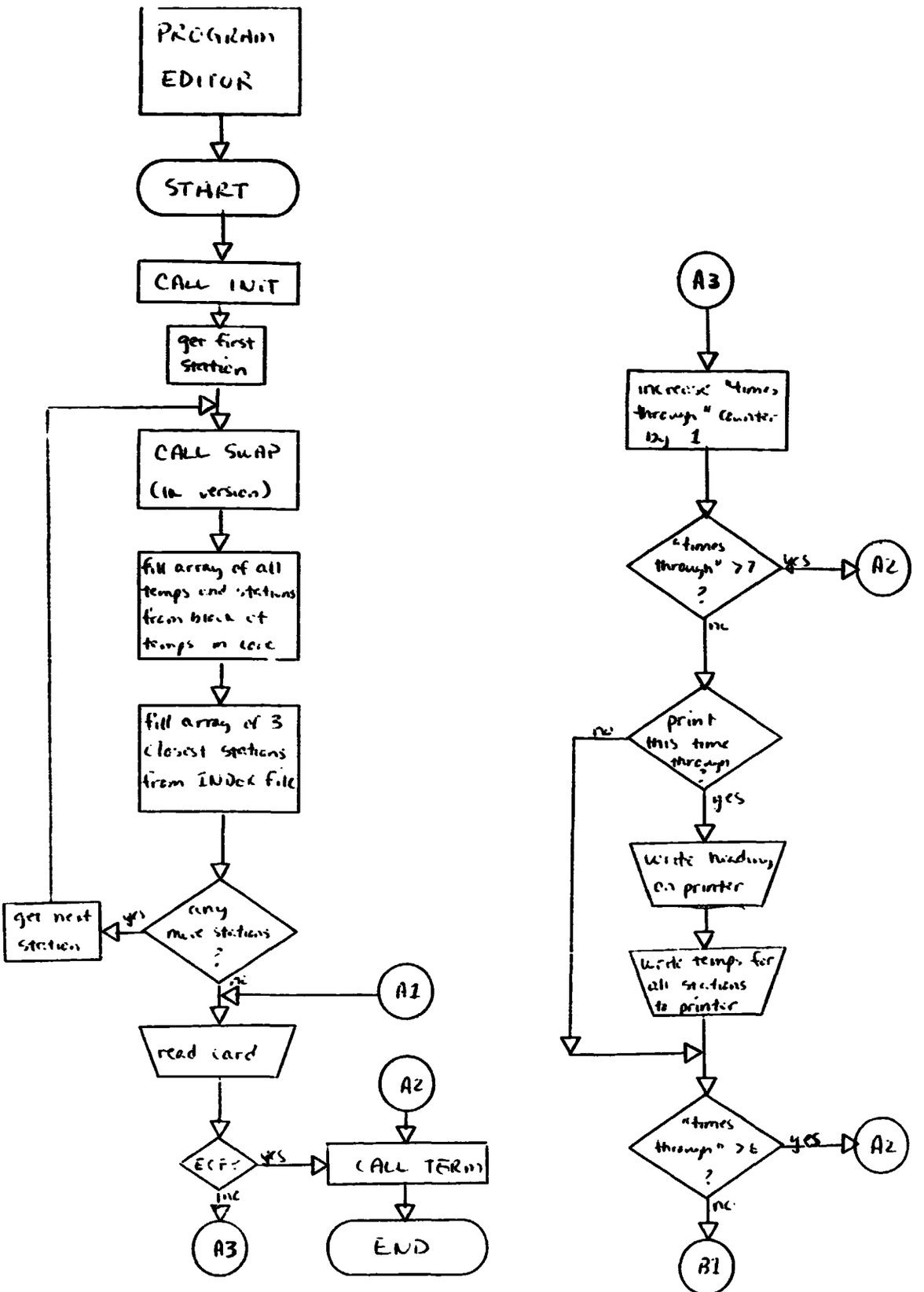
file CROPDATA. EDITOR then reads a card, obtaining values for variables IP, IC, IE, and IS. Each time a card is read, EDITOR increments the variable IL and checks for end-of-file. The edit process is executed three times. The first time, the contents of CROPDATA are listed; CHECK is called to identify values that need to be replaced; REPLAC is called and messages are written to the printer. The second execution of the edit process calls both CHECK and REPLAC. During the third pass through the program, CHECK is called and CROPDATA is rewritten from XTEMP. TERM is called after the completion of the third execution.

3.2.12.6 Flowchart

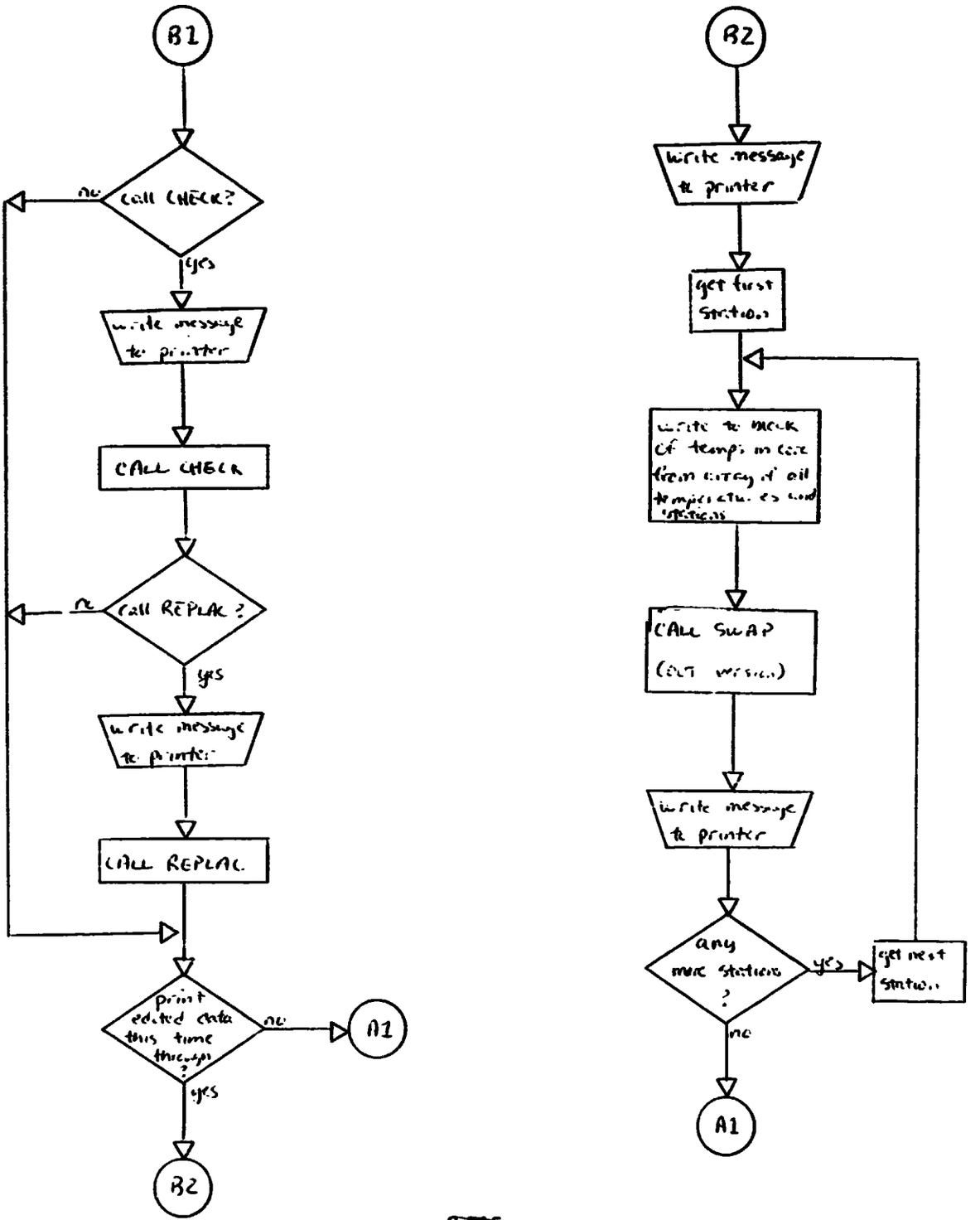
Next page.

3.2.12.7 Listing

Follows flowchart.


PROGRAM EDITOR
START
CALL INIT
get first station
CALL SWAP (IN version)
fill array of all temps and stations from block of temps on core
fill array of 3 closest stations from INDEX file
any more stations ?
yes
get next station
no
A1
read card
EOF?
yes
no
A2
CALL TERM
END
A3
A3
increase "times through" counter by 1
"times through" > 7 ?
yes
A2
no
print this time through ?
no
yes
write heading on printer
write temps for all stations to printer
"times through" > 6 ?
yes
A2
no
A1

PROGRAM EDITOR, CONT



```
PROGRAM EDITOR
C  MAIN PROGRAM FOR CROP CALENDAR EDITING
      REAL XTEMP(511,14,3),XHAR(511,3),XSD(511,3)
      INTEGER LOWR(511,3),HIGHR(511,3),NMO(511),TEMPS(14,2),STATN(511,3)
      INTEGER ITYPE(3)/'MIN ','MAX ','DIF '/
      INTEGER IU/1/,JOUT/2/,JULIAN,IP,IC,IE,IS,IL
      INTEGER*2 NT(511,3),IAP(511,3)
      INTEGER*2 NUMSTA,FILLER
      LOGICAL*1 IFLAG(511,15,2)
      COMMON /INDEX/ NUMSTA,FILLER,NMO
      COMMON /VALI/ XTEMP,XHAR,XSD,LOWR,HIGHR,NT,IAP,JULIAN,IFLAG
      COMMON /CLOSE/STATN
      CALL INIT
      DO 50 K = 1,NUMSTA
      CALL SWAP(IN,NMO(K),TEMPS,JULIAN)
      DO 30 I = 1,3
      IAP(K,I)=0
      DO 10 J = 1,NUMSTA
      IF (NMO(J).EQ.STATN(K,I)) GOTO 20
   10 CONTINUE
      GOTO 30
   20 IAP(K,I)=J
   30 CONTINUE
      DO 40 J = 1,14
      DO 40 I = 1,2
      IF (TEMPS(J,I).EQ.999) TEMPS(J,I)=9999
      XTEMP(K,J,I)=TEMPS(J,I)
   40 CONTINUE
   50 CONTINUE
      IL=0
      READ(5,20001,  IO=9000) IP,IC,IE,IS
20001 FORMAT(2014)
      IL=IL+1
      IF (IL.GT.7)  GOTO 9000
      IF (IP.NE.1) GOTO 100
      WRITE(6,10001) IL
10001 FORMAT('1  CROP CALANDER DATA LISTER -- PASS',I3)
      DO 90 K = 1,NUMSTA
      WRITE(6,10002)
10002 FORMAT(' ')
      DO 70 I = 1,2
      J=3-I
      WRITE(6,10003) NMO(K),ITYPE(J),JULIAN,(XTEMP(K,L,J),L=1,14)
10003 FORMAT(1X,I5,1X,A4,I3,14F5.0)
   70 CONTINUE
   90 CONTINUE
  100 IF (IL.GT.6) GOTO 9000
      IF (IC.LT.1) GOTO 300
      WRITE(6,10004) IL
10004 FORMAT('  CROP CALANDER CHECKER  -- PASS',I3)
      CALL CHECK(IC,IL)
      IF (IE.LT.1) GOTO 300
      WRITE(6,10005) IL
10005 FORMAT('  CROP CALANDER EDITOR  -- PASS',I3)
      CALL REPLAC(IE)
  300 IF (IS.NE.1) GOTO 60
      WRITE(6,10006) IL
10006 FORMAT('  CROP CALANDER REWRITE FILE  -- PASS',I3)
      DO 320 K = 1,NUMSTA
      DO 310 I = 1,2
      DO 310 J = 1,14
      TEMPS(J,I)=XTEMP(K,J,I)+0.5
  310 CONTINUE
```

84

```
      CALL SWAP(OUT,WMO(K),TEMPS,JULIAN)
C     WRITE(5,10007) WMO(K)
10007 FORMAT('0',I5,' REWRITTEN TO FILE')
  320 CONTINUE
      GOTO 60
 6000 CALL TERM
      STOP
      END
```

64

3.2.13 JCL FILE EDIT

Executing JCL file EDIT invokes the EDITOR program.

3.2.13.1 Linkages

None.

3.2.13.2 Interfaces

EDIT should not be executed until CAPTURE has successfully filled file CROPDATA.

3.2.13.3 Inputs

Direct card input to this file is necessary. See Section 4.2.2.3 for further discussion and input format.

3.2.13.4 Outputs

None.

3.2.13.5 Description

PROC EDIT binds the files referenced by program EDITOR to that program and requests the invocation of EDITOR.

3.2.13.6 Listing

```
//EDIT⌀PROC
//EDIT⌀EXEC⌀PGM=EDITOR,TIME=(,15)
//STEPLIB⌀DD⌀DSN=W.EDS.CCEA.PHASE3.LOAD,DISP=SHR
//FT02F001⌀DD⌀DSN=W.EDS.CCEA.{GROUPI  }.INDEX,DISP=SHR
                              {GROUPII }
//FT04F001⌀DD⌀DSN=W.EDS.CCEA.{GROUPI  }.CROPDATA,DISP=SHR
                              {GROUPII }
//FT05F001⌀DD⌀DUMMY
//FT06F001⌀DD⌀SYSOUT=A
//⌀⌀⌀⌀⌀PEND
```

3.2.14 SUBROUTINE CHECK

CHECK determines which data in CROPDATA should be replaced.

3.2.14.1 Linkages

CHECK uses COMMON block INDEX and VAL1.

3.2.14.2 Interfaces

None.

3.2.14.3 Inputs

IC = 1 do not print results;
= 2 print results

IL the number of times through the edit procedure for this station.

XTEMP(I,14,3) an array containing the information on file CROPDATA, as well as space for the daily differences between maximum and minimum temperature, where I is the number of crop calendar stations.

IAP(I,3) an array containing the positions in array WMO of the three closest neighbor stations for each crop calendar station, where I is the number of stations.

3.2.14.4 Outputs

Outputs from CHECK contained in COMMON block VAL1 and where I is the number of stations involved:

XBAR(I,3) mean of each station's max,min temperatures and their differences.

XSD(I,3) standard deviations of each station's three temperature types.

LOWB(I,3) lower confidence bounds of each station's three temperature types.

HIGHB(I,3) upper confidence bounds of each station's three temperature types.

NT(I,3) number of acceptable values of each station's three temperature types.

TFLAG(I,15,2) a logical array indicating "good" or "bad" temperatures; the 15th position is flagged "good" if all values for that temp. type are "good" and "bad" if at least one temp. is "bad".

CDAY(14,2) array for printed output indicating values that should be manually checked and edited.

3.2.14.5 Description

CHECK begins by counting reasonable temperatures and flagging unreasonable ones. When a station has at least 5 "good" max temps. and/or 5 "good" min temps. for the two-week period, the mean temperature, variance, standard deviation, and upper and lower confidence bounds of the temperature type are calculated. When both max and min are present for a day, differences between the two are calculated and summed; provided at least 5 differences for a station exist, the same statistics as for the max and/or min are calculated. CHECK attempts to find and calculate similar data for a station's three closest neighbor stations; 4-station averages of the mean temperature, standard deviation, lower and upper confidence bounds are figured. Whenever a large variation occurs between the mean of a temperature type and the corresponding 4-station mean, the 4-station mean is substituted.

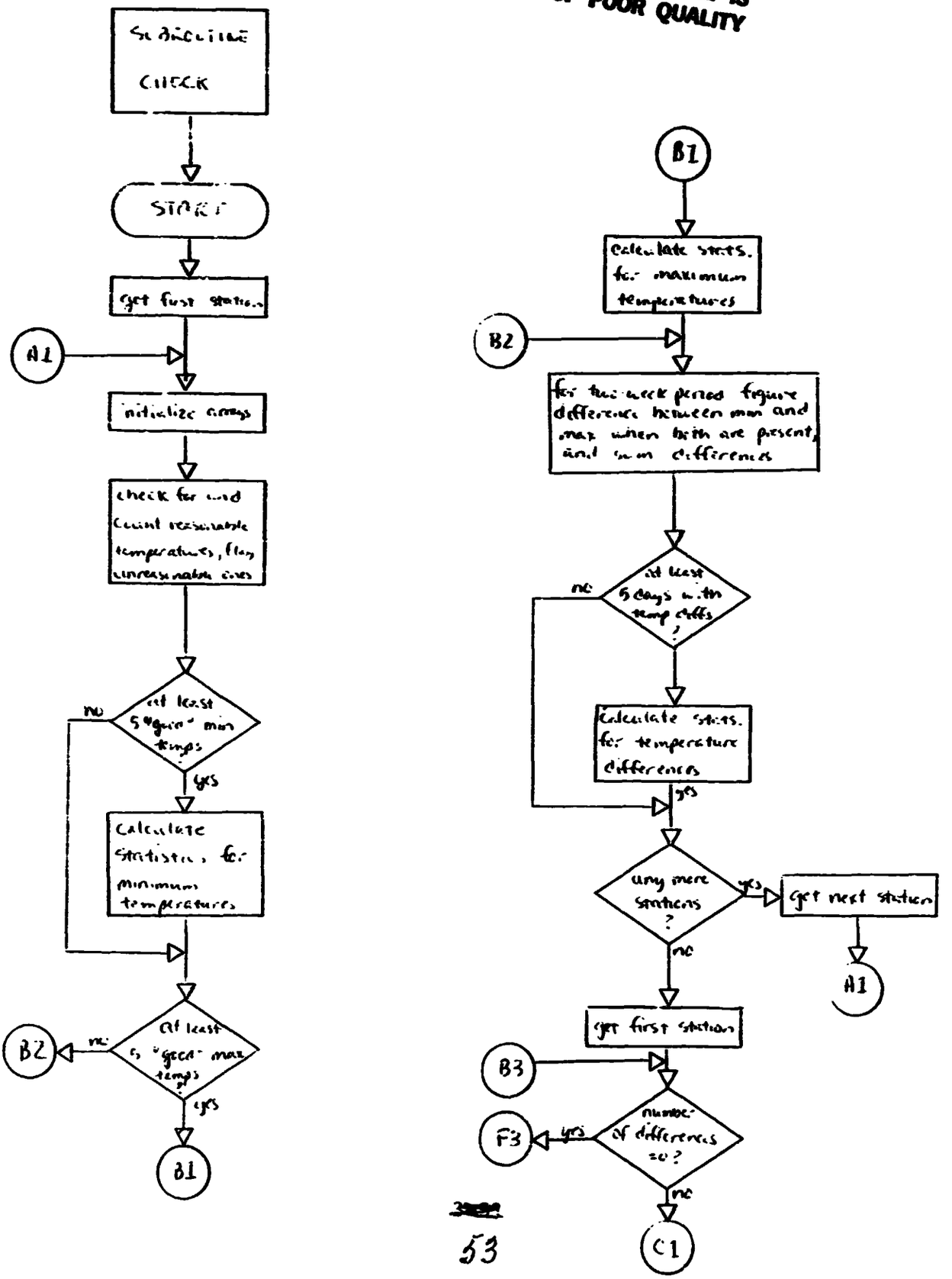
Temperatures falling outside of the upper and lower confidence bounds are flagged. Then the maximum of one day is compared to the minimum of the next; whenever too much or no variation between the two are encountered, temperatures are flagged according to the type of variation. After all temperatures for a particular station have been checked, an array CDAY, denoting acceptable vs. not acceptable temperatures is filled. Beginning with the checks outside the upper and lower confidence bounds, the entire process is repeated. After the second pass through this part of CHECK, the value of IC is checked. For IC=1, no printout is generated, but for IC=2, the following is printed: a listing of max and min temperatures and their differences, the number of values of each, the mean, standard deviation, and lower and upper confidence bounds of each temperature, as well as the array CDAY indicating values that should be manually checked and edited before procedure SUZYQ is run.

3.2.14.6 Flowchart

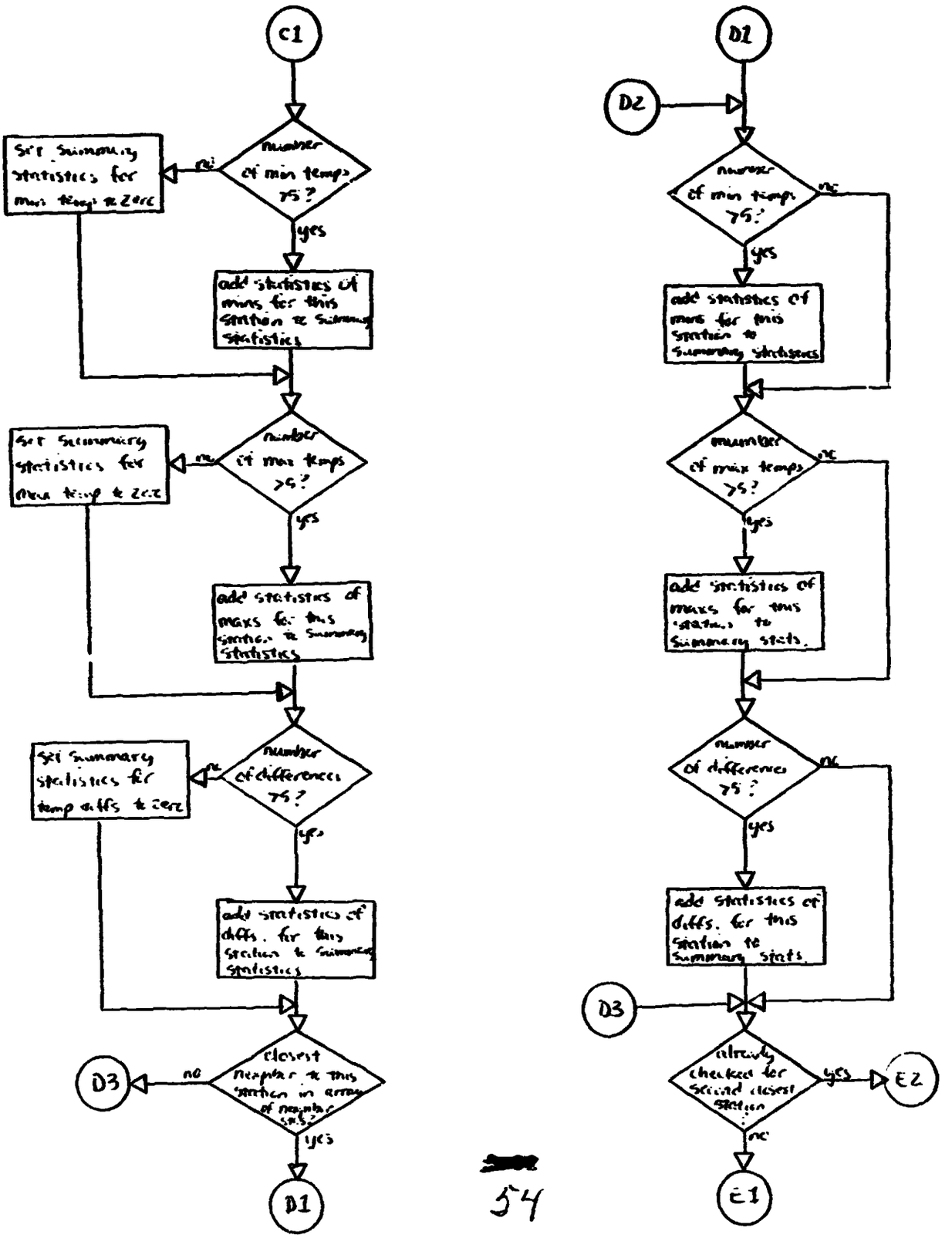
Next page.

3.2.14.7 Listing

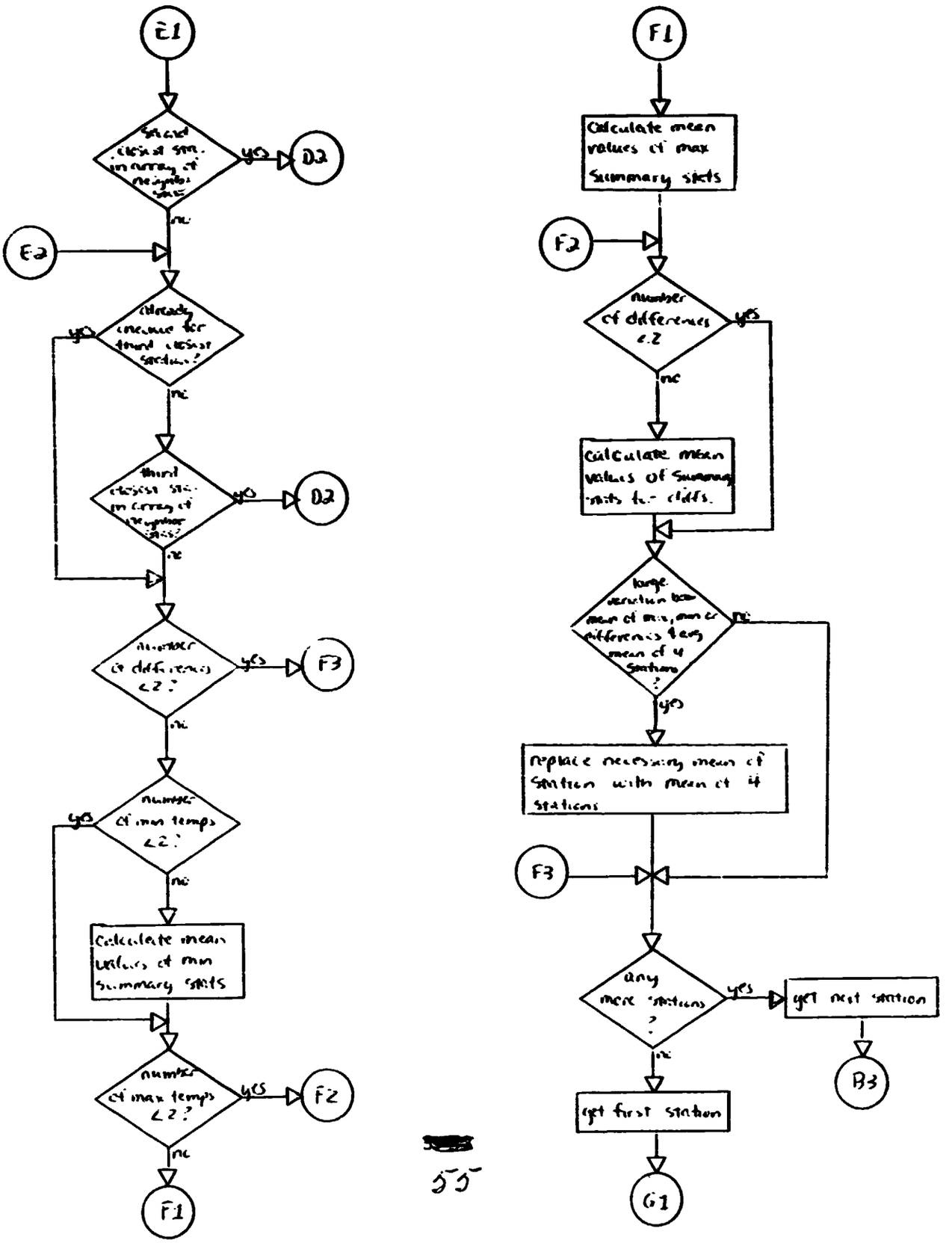
Follows flowchart.



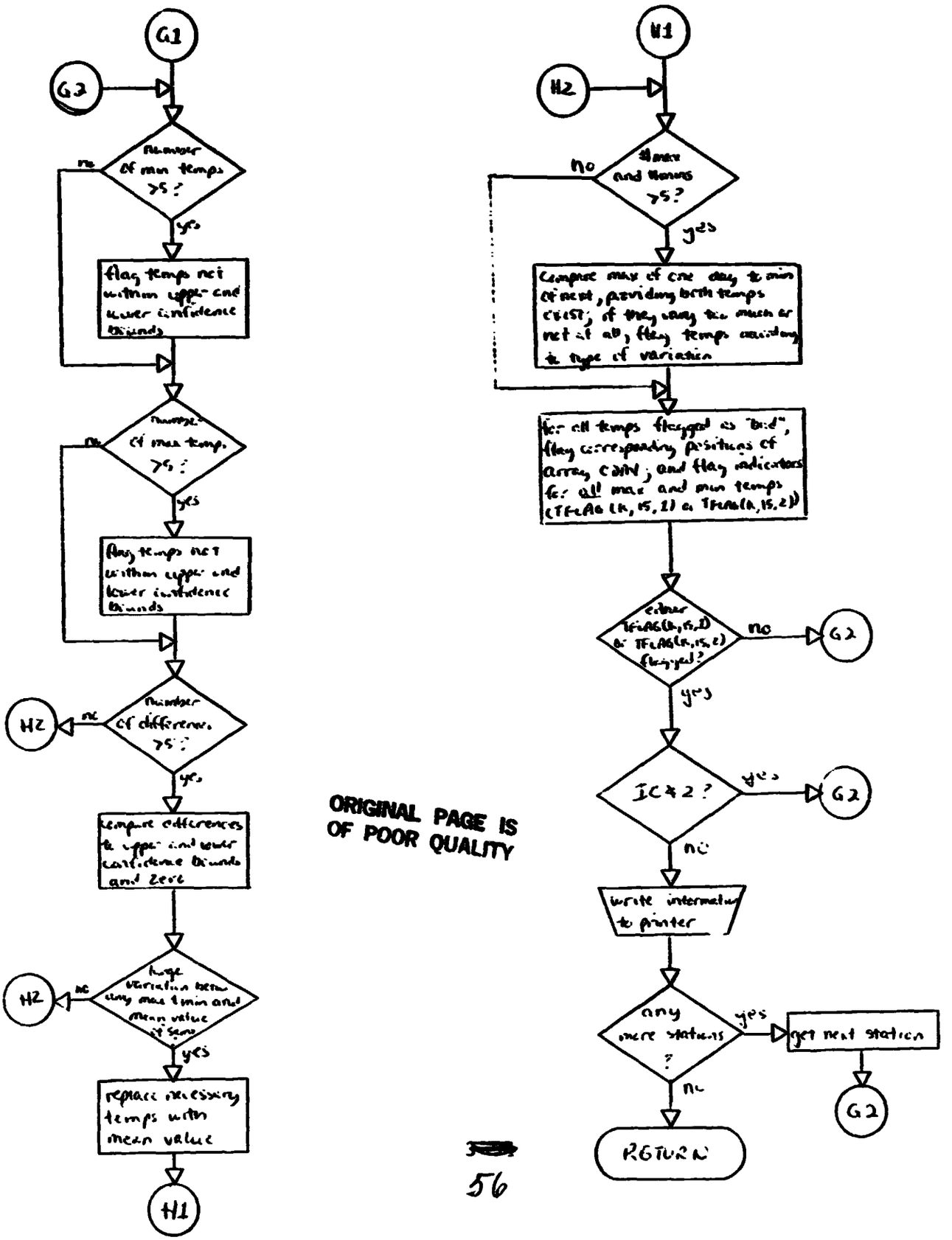
SUBROUTINE CHECK, CON'T



SUBROUTINE CHECK, CONT



SUBROUTINE CHECK, CONT



ORIGINAL PAGE IS OF POOR QUALITY

```
      SUBROUTINE CHECK(IC,IL)
      REAL XTEMP(5,1,14,3),XHAR(5,1,3),XSD(5,1,3),ABAR(3),ASD(3),RN(3),
     * SX(3)
      DATA ZK1(5)/1.645,1.96,2.326,2.748,3.291,3.891/
      DATA ZK2(5)/1.96,2.326,2.748,3.291,3.891,4.417/
      DATA Z0,Z1,ZM,ZR,X2
      INTEGER LOWA(5,1,3),HIGHA(2,1,3),NQ(5,1,1),TEMPS(14,2),CDAY(14,2),
     * NA(3),LOWA(3),HIGHA(3)
      INTEGER ITYPE(3)/'MIN ','MAX ','DIF '/
      INTEGER HIGH/110/,LOW/-50/,CXX/' XX '/,BLANK/'     '/,JULIAN,
     * JULL,IL,IC
      INTEGER*2 NT(5,1,3),IAP(5,1,3)
      INTEGER*2 NUMSTA,FILLER
      LOGICAL*1 TFLAG(5,1,15,2)
      COMMON /INDEX/ NUMSTA,FILLER,WMO
      COMMON /VALI/ XTEMP,XHAR,XSD,LOWH,HIGHB,NT,IAP,JULIAN,TFLAG
      DO 70 K = 1,NUMSTA
      DO 10 I = 1,3
      RN(I)=0.0
      SX(I)=0.0
      XHAR(K,I)=0.0
      XSD(K,I)=0.0
      NT(K,I)=0
      LOWH(K,I)=0
      HIGHH(K,I)=0
   10 CONTINUE
      DO 20 I = 1,2
      TFLAG(K,15,I)=.FALSE.
      DO 14 J = 1,14
      TFLAG(K,J,I)=.TRUE.
      TEMPS(J,I)=XTEMP(K,J,I)+0.5
      IF (TEMPS(J,I)-HIGH) 12,16,16
   12 IF (TEMPS(J,I)-LOW) 14,16,16
   14 RN(I)=RN(I)+1.0
      SX(I)=SX(I)+XTEMP(K,J,I)
      TFLAG(K,J,I)=.FALSE.
   16 CONTINUE
   20 CONTINUE
      DO 55 I = 1,2
      IF (RN(I)-5.0) 55,52,52
   52 XHAR(K,I)=SX(I)/RN(I)
      NT(K,I)=RN(I)
      XR=0.0
      DO 54 J = 1,14
      IF (TFLAG(K,J,I)) GOTO 54
      X2=XTEMP(K,J,I)-XHAR(K,I)
      XR=XR+X2*X2
   54 CONTINUE
      XSD(K,I)=SQRT(XR/(RN(I)-1.0))
      ZA=XSD(K,I)*ZK(IL)
      LOWH(K,I)=XHAR(K,I)-ZA
      HIGHH(K,I)=XHAR(K,I)+ZA+0.5
   55 CONTINUE
      DO 62 J = 1,14
      IF ((TFLAG(K,J,1)).AND.(TFLAG(K,J,2))) GOTO 62
      RN(3)=RN(3)+1.0
      XTEMP(K,J,3)=TEMPS(J,2)-TEMPS(J,1)
      SX(3)=SX(3)+XTEMP(K,J,3)
   62 CONTINUE
      IF (RN(3)-5.0) 70,64,64
   64 XHAR(K,3)=SX(3)/RN(3)
      NT(K,3)=RN(3)
      XR=0.0
```

```
      DO 66 J = 1,14
      IF ((IFLAG(K,J,1)).OR.(IFLAG(K,J,2))) GOTO 66
      XZ=XTEMP(K,J,3)-XBAR(K,3)
      XR=XR+XZ*XZ
   66 CONTINUE
      XSD(K,3)=SQRT(XR/(RN(3)-1.0))
      ZA=XSD(K,3)*ZKP(IL)
      LOWR(K,3)=XHAR(K,3)-ZA
      HIGHR(K,3)=XHAR(K,3)+ZA*0.5
   70 CONTINUE
      DO 100 K = 1,NUMSTA
      IF (NT(K,3).EQ.0) GOTO 100
      DO 76 I = 1,3
      IF (NT(K,I)-5) 74,72,72
   72 IA(I)=1
      AHAR(I)=XHAR(K,I)
      ASD(I)=XSD(K,I)
      LOWA(I)=LOWR(K,I)
      HIGHA(I)=HIGHR(K,I)
      GOTO 76
   74 IA(I)=0
      AHAR(I)=0.0
      ASD(I)=0.0
      LOWA(I)=0
      HIGHA(I)=0
   76 CONTINUE
      DO 86 I = 1,3
      IF (IAM(K,I).LT.1) GOTO 86
      J=IAM(K,I)
      DO 84 IM = 1,3
      IF (NT(J,IM)-5) 84,82,82
   82 NA(IM)=NA(IM)+1
      AHAR(IM)=AHAR(IM)+XHAR(J,IM)
      ASD(IM)=ASD(IM)+XSD(J,IM)
      LOWA(IM)=LOWA(IM)+LOWR(J,IM)
      HIGHA(IM)=HIGHA(IM)+HIGHR(J,IM)
   84 CONTINUE
   86 CONTINUE
      IF (NA(3).LT.2) GOTO 100
      DO 88 I = 1,3
      IF (NA(I).LT.2) GOTO 88
      AHAR(I)=AHAR(I)/FLOAT(NA(I))
      ASD(I)=ASD(I)/FLOAT(NA(I))
      LOWA(I)=LOWA(I)/NA(I)
      HIGHA(I)=HIGHA(I)/NA(I)
   88 CONTINUE
      ZA=XHAR(K,3)-AHAR(3)
      IF (ZA+ASD(3)) 90,100,100
   90 I2=ABS(XHAR(K,2)-AHAR(2))
      I1=ABS(XHAR(K,1)-AHAR(1))
      IF (I2-I1) 92,96,96
   92 I=1
      J=1
      GOTO 98
   94 I=1
      J=2
      GOTO 98
   96 I=2
      J=2
   98 DO 99 IM = I,J
      XHAR(K,IM)=AHAR(IM)
      XSD(K,IM)=ASD(IM)
      LOWR(K,IM)=LOWA(IM)
      HIGHR(K,IM)=HIGHA(IM)
   99 CONTINUE
      XHAR(K,3)=AHAR(3)
      XSD(K,3)=ASD(3)
      LOWR(K,3)=LOWA(3)
```

```
      HIGHB(K,3)=HIGHA(3)
  100 CONTINUE
      DO 200 K = 1,NUMSTA
      DO 110 I = 1,2
      IF (NT(K,I)-5) 110,102,102
  102 DO 108 J = 1,14
      IF (TFLAG(K,J,I)) GOTO 108
      IT=XTEMP(K,J,I)+0.5
      IF (IT-LOWA(K,I)) 106,104,104
  104 IF (IT-HIGHA(K,I)) 108,108,106
  106 TFLAG(K,J,I)=.TRUE.
  108 CONTINUE
  110 CONTINUE
      IF (NT(K,3)-5) 130,112,112
  112 DO 126 J = 1,14
      IF (TFLAG(K,J,1).OR.TFLAG(K,J,2)) GOTO 126
      IZ=XTEMP(K,J,3)
      IF (IZ-HIGHA(K,3)) 114,114,118
  114 IF (IZ-0) 118,118,116
  116 IF (IZ-LOWA(K,3)) 118,126,126
  118 IY=ABS(XTEMP(K,J,2)-XBAR(K,2))
      IO=ABS(XTEMP(K,J,1)-XBAR(K,1))
      IF (IO-IY) 120,122,124
  120 TFLAG(K,J,2)=.TRUE.
      GOTO 126
  122 TFLAG(K,J,2)=.TRUE.
      TFLAG(K,J,1)=.TRUE.
      GOTO 126
  124 TFLAG(K,J,1)=.TRUE.
  126 CONTINUE
  130 IF ((NT(K,1).LT.5).OR.(NT(K,2).LT.5)) GOTO 140
      DO 138 J = 2,14
      I=J-1
      IF (TFLAG(K,J,1).OR.TFLAG(K,I,2)) GOTO 138
      IT=XTEMP(K,I,2)+0.5
      IZ=XTEMP(K,J,1)+0.5
      IF (IT-IZ) 132,135,134
  132 IY=ABS(XTEMP(K,I,2)-XBAR(K,2))
      IO=ABS(XTEMP(K,J,1)-XBAR(K,1))
      IF (IO-IY) 134,135,137
  134 TFLAG(K,I,2)=.TRUE.
      GOTO 138
  135 TFLAG(K,J,1)=.TRUE.
      TFLAG(K,I,2)=.TRUE.
      GOTO 138
  137 TFLAG(K,J,1)=.TRUE.
  138 CONTINUE
  140 DO 150 I = 1,2
      DO 144 J = 1,14
      CDAY(J,I)=BLANK
      IF (.NOT.TFLAG(K,J,I)) GOTO 144
      CDAY(J,I)=C*X
      TFLAG(K,15,I)=.TRUE.
  144 CONTINUE
  150 CONTINUE
      IF ((.NOT.TFLAG(K,15,1)).AND.(.NOT.TFLAG(K,15,2))) GOTO 200
      IF (IC.EQ.2) GOTO 200
      WRITE(6,10012)
10012 FORMAT('0')
      DO 160 I = 1,2
      J=3-I
      WRITE(6,10015) WMO(K),ITYPE(J),JULIAN,(XTEMP(K,IM,J),IM=1,14),
     & NT(K,J),XBAR(K,J),XSD(K,J),LOWB(K,J),HIGHB(K,J),
     & (CDAY(IM,J),IM=1,14)
10015 FORMAT(1X,I5,1X,A4,1X,I3,14F6.0,I5,2F8.2,2I5/' CHECK',9X,14(2X,A4))
  160 CONTINUE
      WRITE(6,10015) WMO(K),ITYPE(3),JULIAN,(XTEMP(K,IM,3),IM=1,14),
     & NT(K,3),XBAR(K,3),XSD(K,3),LOWA(K,3),HIGHA(K,3)
```

```
10015 FORMAT(1X,I6,1X,A4,I3,14F6.0,I5,2F8.2,2I5)
  200 CONTINUE
      RETURN
      END
```

3.2.15 SUBROUTINE REPLAC

REPLAC attempts to find and insert replacement values for temperatures flagged as unacceptable by CHECK.

3.2.15.1 Linkages

REPLAC uses COMMON blocks INDEX and VAL1.

3.2.15.2 Interfaces

None.

3.2.15.3 Inputs

IE = 1 edit but do not print results
= 2 edit and print results
JULIAN ending date of the previous 2-week period

where I is the number of stations involved,

XTEMP(I,14,3)	IAP(I,3)
XBAR(I,3)	XSD(I,3)
LOWB(I,3)	HIGHB(I,3)
NT(I,3)	TFLAG(I,15,2)

See sections 3.2.14.3 and 3.2.14.4 for descriptions of these variables.

3.2.15.4 Outputs

Replacement values inserted into array XTEMP (I,14,3), where I is the number of stations, as well as messages indicating a successful substitution or the inability to make a substitution.

3.2.15.5 Description

This subroutine is executed twice each time it is called. REPLAC determines if any values for a particular station need to be replaced. When substitutions need to be made, several algorithms are available:

A) At least seven temperatures of the station and type in

question are available, as well as data from alternate stations;

B) at least seven temps are available, but no alternate station data;

C) fewer than seven temps are available.

Algorithm A will utilize up to four days of data from the primary station and up to five days of data from each alternate station, provided the stations also meet the criterion of at least seven "good" temperatures. For each station involved, the temperatures are standardized and averaged. A weighting factor is also calculated for each station, as well as an average weighting factor. The replacement value is the sum of the standardized value for the primary station and the average weighting factor.

Algorithm B will first attempt to find a replacement value using the other temperature of the day if it is available (i.e., the max to find the min or vice versa). If this is not successful, REPLAC will try to use the average of up to four days of data for the replacement temperature. If no alternate days are available within a prescribed vicinity of the day in question, the replacement value will be the mean max or min value of the entire 14-day period.

Algorithm C will use the average temperature of the alternate stations for the day in question as the replacement. If Algorithm C does not generate a replacement value, a message to that effect will be printed.

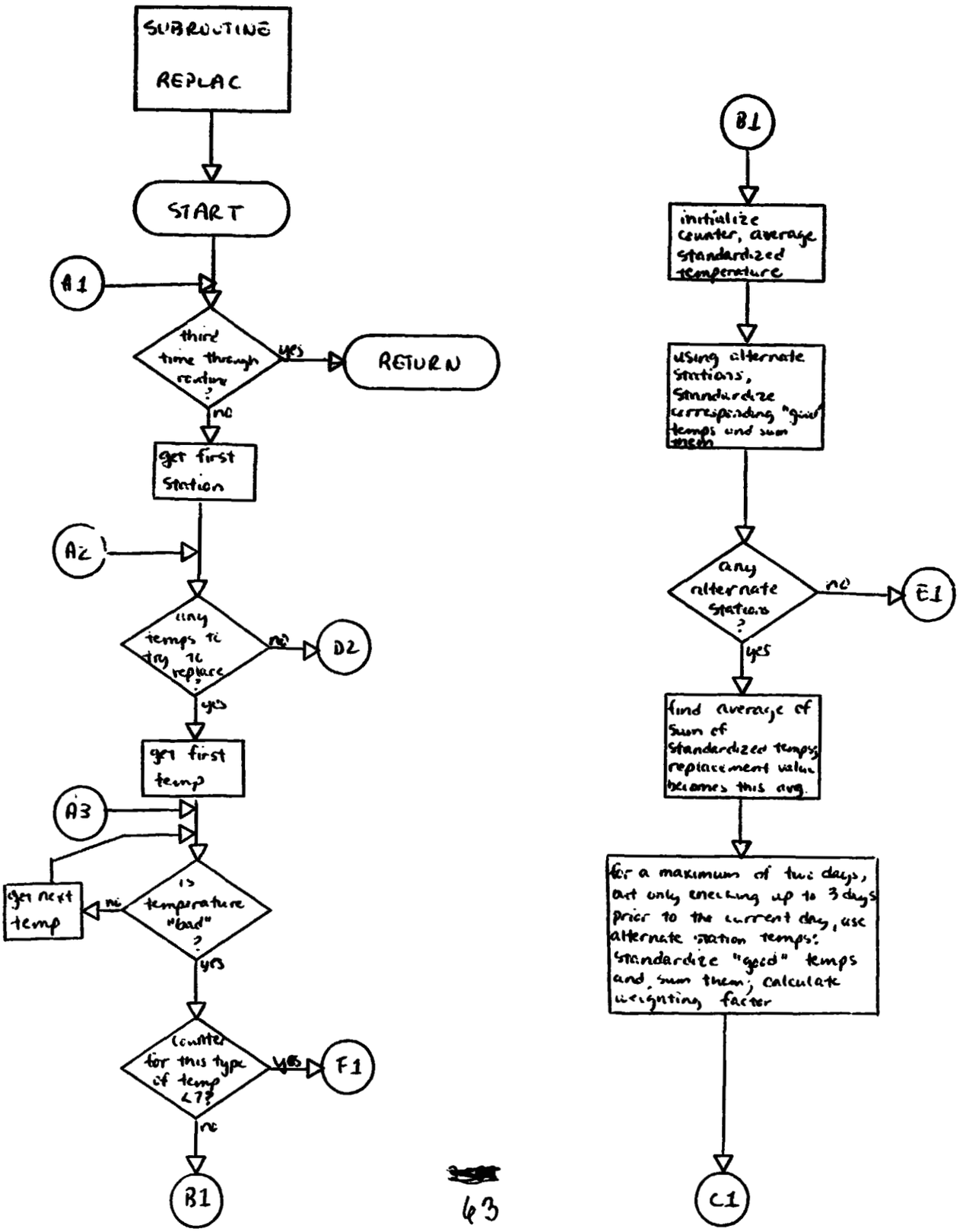
After substitutions for all stations needing them have been made and the subroutine has been executed twice, control returns to program EDITOR.

3.2.15.6 Flowchart

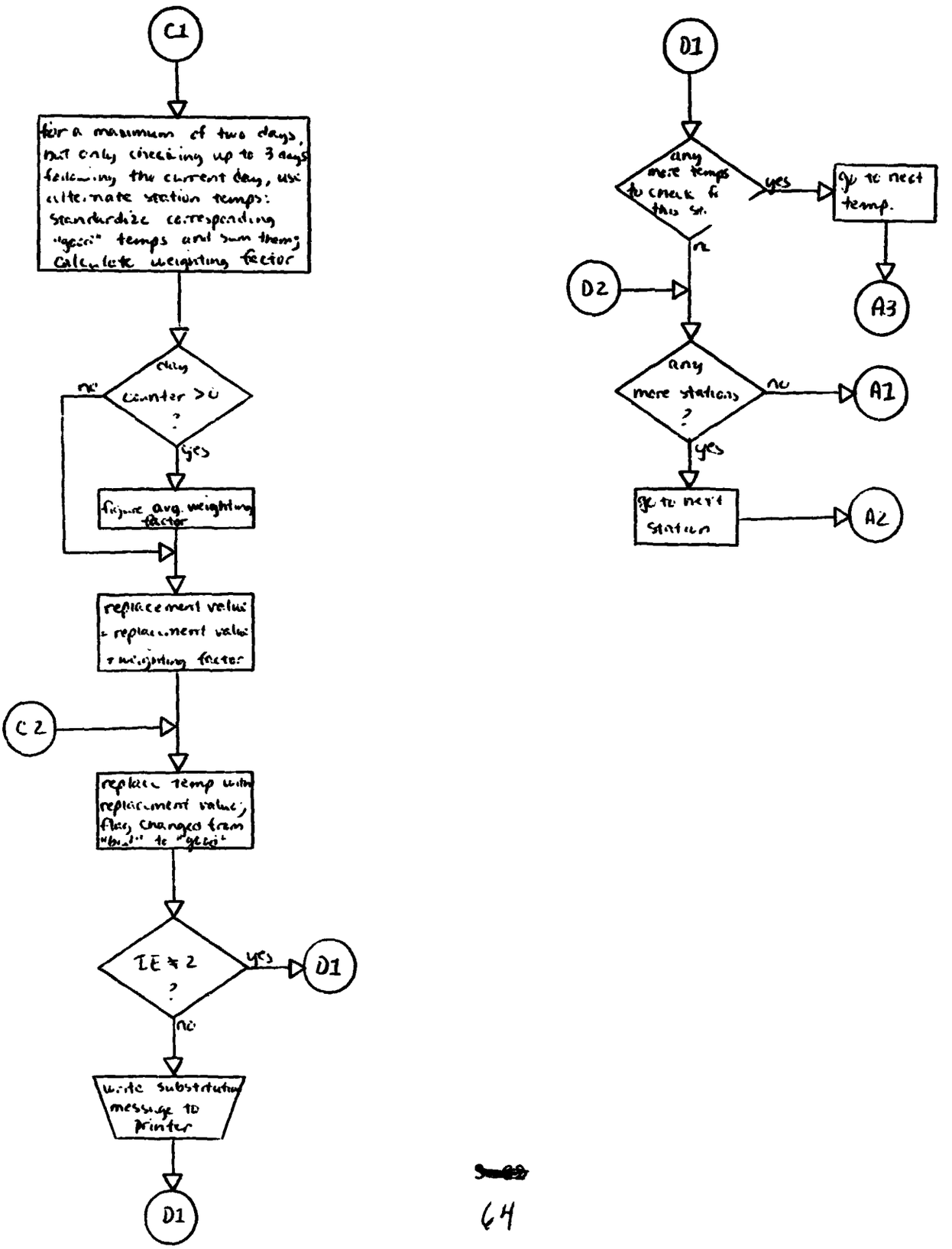
Next page.

3.2.15.7 Listing

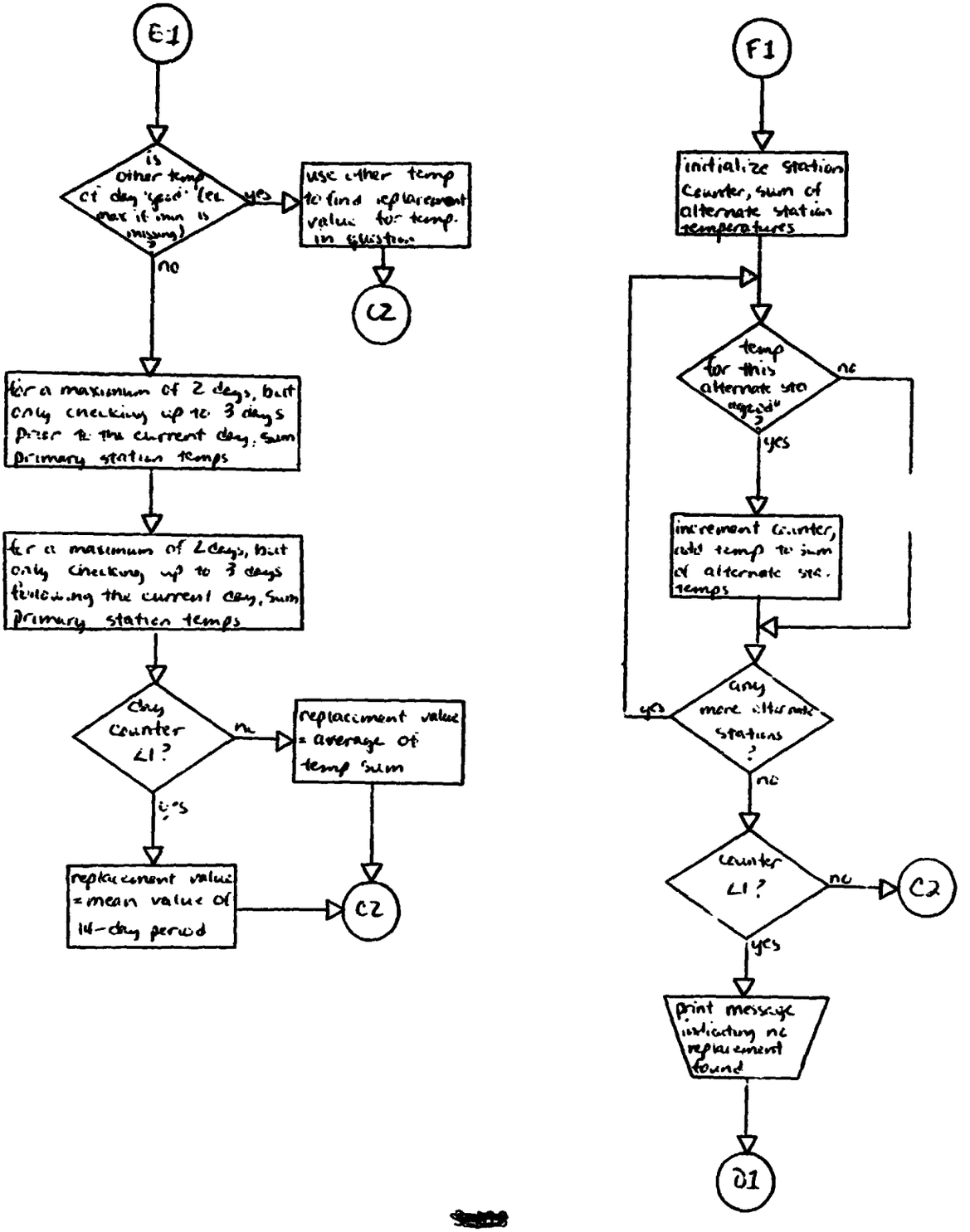
Follows flowchart.


SUBROUTINE
REPLAC
START
A1
third time through routine ?
yes
RETURN
no
get first station
A2
any temps to try to replace ?
no
D2
yes
get first temp
A3
get next temp
no
is temperature "bad" ?
yes
counter for this type of temp < 7 ?
yes
F1
no
B1
B1
initialize counter, average standardized temperature
using alternate stations, standardize corresponding "good" temps and sum them
any alternate stations ?
no
E1
yes
find average of sum of standardized temps; replacement value becomes this avg.
for a maximum of two days, but only checking up to 3 days prior to the current day, use alternate station temps; standardize "good" temps and sum them; calculate weighting factor
C1

SUBROUTINE REPLAC, CON·T



SUBROUTINE REPLAC, CONT



3.3.15.7
SUBROUT

```
      SUBROUTINE REPLACE(IE)
      REAL ATEMP(511,14,3),XBAR(511,3),XSD(511,3),SX(3),ON(3)
      REAL ZO,Z1,Z2,A2,X2
      INTEGER LOWT(511,3),HIGHT(511,3),WMO(511),TEMPS(14,2)
      INTEGER ITYPE(3)/'MIN ','MAX ','DIF '/
      INTEGER JULIAN,JULL,IF
      INTEGER*2 NT(511,3),IAP(511,3),NALT(2)
      INTEGER*2 NUMSTA,FILLER
      LOGICAL*1 TFLAG(511,15,2)
      COMMON /INDEX/ NUMSTA,FILLER,WMO
      COMMON /VAL1/ ATEMP,XBAR,XSD,LOWT,HIGHT,NT,IAP,JULIAN,TFLAG
      DO 5000 KK = 1,2
      DO 4000 ISTA = 1,NUMSTA
      IF ((.NOT.TFLAG(ISTA,15,1)).AND.(.NOT.TFLAG(IS   ,15,2))) GOTO 4000
      DO 3000 JDAY = 1,14
      DO 3000 IM = 1,2
      IF (.NOT.TFLAG(ISTA,JDAY,IM)) GOTO 3000
      IF (NT(ISTA,IM).LT.7) GOTO 2200
C
CCC   ** REPLACE MISSING VALUE USING MEANS AND STANDARD DEVIATIONS
CCC      CHECK FOR ALTERNATE STATIONS
CCC
      Z0=0.0
      N=0
      DO 2040 I = 1,3
      IF (IAP(ISTA,I).LT.1) GOTO 2040
      K=IAP(ISTA,I)
      IF (TFLAG(K,JDAY,IM)) GOTO 2040
      IF (NT(K,IM).LT.7) GOTO 2040
      N=N+1
      NALT(N)=K
      Z0=Z0+(ATEMP(K,JDAY,IM)-XBAR(K,IM))/XSD(K,IM)
 2040 CONTINUE
      IF (N.LT.1) GOTO 2110
C
CCC   ** ALTERNATE STATIONS AVAILABLE
CCC
      Z0=Z0/FLOAT(N)
      X0=Z0*XSD(ISTA,IM)+XBAR(ISTA,IM)
CCC
CCC      CHECK FOR ALTERNATE STATION BIAS
CCC
      JDAY=IDAY
      Z1=0.0
      N2=0
 2050 JDAY=JDAY-1
      IF ((JDAY.LT.1).OR.(JDAY.LT.(IDAY-3))) GOTO 2070
      IF (TFLAG(ISTA,JDAY,IM)) GOTO 2050
      Z1=0.0
      N1=0
      DO 2060 I = 1,N
      K=NALT(I)
      IF (TFLAG(K,JDAY,IM)) GOTO 2060
      N1=N1+1
      Z1=Z1+(ATEMP(K,JDAY,IM)-XBAR(K,IM))/XSD(K,IM)
 2060 CONTINUE
      IF (N1.LT.1) GOTO 2065
      X2=Z1/FLOAT(N1)*XSD(ISTA,IM)+XBAR(ISTA,IM)
      Z2=Z2+(ATEMP(ISTA,JDAY,I)-X2)
      N2=N2+1
 2065 IF (N2.LT.2) GOTO 2050
 2070 JDAY=JDAY-
 2080 JDAY=JDAY+1
```

```
      IF ((JDAY.GT.14).OR.(JDAY.GT.(IDAY+3))) GOTO 2100
      IF (IFLAG(ISTA,JDAY,IM)) GOTO 2090
      Z1=0.0
      N1=0
      DO 2090 I = 1,N
      K=NALT(I)
      IF (IFLAG(K,JDAY,IM)) GOTO 2090
      Z1=Z1+(ATEMP(K,JDAY,IM)-XBAR(K,IM))/XSD(K,IM)
      N1=N1+1
 2090 CONTINUE
      IF (N1.LT.1) GOTO 2095
      XZ=Z1/FLOAT(N1)*XSD(ISTA,IM)+XBAR(ISTA,IM)
      ZH=ZH+(ATEMP(ISTA,JDAY,IM)-XZ)
      NZ=NZ+1
 2095 IF (NZ.LT.4) GOTO 2080
 2100 IF (NZ.GT.0) ZH=ZH/FLOAT(NZ)
      YB=YB+ZH
      GOTO 2500
C
C     ** NO ALTERNATE STATIONS AVAILABLE
C        CHECK FOR OTHER TEMP WITHIN PRIMARY STATION
C
 2110 I0=1
      IF (IM.EQ.1) I0=2
      IF (NT(ISTA,I0).LT.5) GOTO 2120
      IF (IFLAG(ISTA,JDAY,I0)) GOTO 2120
      Z0=(ATEMP(ISTA,JDAY,I0)-XBAR(ISTA,I0))/XSD(ISTA,I0)
      ZH=Z0*XSD(ISTA,IM)+XBAR(ISTA,IM)
      GOTO 2500
C
C     ** NOT OTHER TEMP AVAILABLE
C        CHECK SAME TEMPS WITHIN PRIMARY STATION
C
 2120 JDAY=IDAY
      Z4=0.0
      NZ=0
 2130 JDAY=JDAY-1
      IF ((JDAY.LT.1).OR.(JDAY.LT.(IDAY-3))) GOTO 2135
      IF (IFLAG(ISTA,JDAY,IM)) GOTO 2130
      Z4=Z4+ATEMP(ISTA,JDAY,14)
      NZ=NZ+1
      IF (NZ.LT.4) GOTO 2130
 2135 JDAY=IDAY
 2140 JDAY=JDAY+1
      IF ((JDAY.GT.14).OR.(JDAY.GT.(IDAY+3))) GOTO 2145
      IF (IFLAG(ISTA,JDAY,IM)) GOTO 2140
      Z4=Z4+ATEMP(ISTA,JDAY,14)
      NZ=NZ+1
      IF (NZ.LT.4) GOTO 2140
 2145 IF (NZ.LT.1) GOTO 2150
      ZH=ZH/FLOAT(NZ)
      GOTO 2500
C
C     ** NO TEMP WITHIN PRIMARY STATINN USEABLE
C        USE PRIMARY STATION MEAN
C
 2150 YB=XBAR(ISTA,IM)
      GOTO 2500
C
C     *** NO STATION MEANS AND STANDARD DEVIATIONS
C        CHECK FOR ALTERNATE STATIONS
C
 2200 Z0=0.0
      N1=0
      DO 2210 I = 1,3
      IF (IAP(ISTA,I).LT.1) GOTO 2210
      K=IAP(ISTA,I)
      IF (IFLAG(K,IDAY,IM)) GOTO 2210
```

```
      N=N+1
      ZO=ZO+XTEMP(K,IDAY,IM)
 2210 CONTINUE
      IF (N.LT.1) GOTO 2600
C
C     ** ALTERNATE STATION AVAILABLE
C
      XR=ZO/FLOAT(N)
C
C     *****     REPLACE MISSING VALUE WITH SUBSITUTE VALUE
C
 2500 XTEMP(ISTA,IDAY,IM)=XR
      TFLAG(ISTA,IDAY,IM)=.FALSE.
      IF (IE.NE.2) GOTO 3000
      JULL=JULIAN+IDAY-1
      WRITE (6,10021) WMO(ISTA),JULL,ITYPE(IM),XR
10021 FORMAT('0STATION',I6,', JULIAN DAY',I4,1X,A4,
     & ' TEMP WAS MISSING. A VALUE OF',F5.0,' HAS BEEN SUBSITUTED.')
      GOTO 3000
C
C     *****  NO VALUE FOUND FOR SUBSITUTION
C
 2600 JULL=JULIAN+IDAY-1
      WRITE(6,10022) WMO(ISTA),JULL,ITYPE(IM)
10022 FORMAT('0*** STATION',I6,', JULIAN DAY',I4,1X,A4,
     & ' TEMP WAS MISSING. NO VALUE AVAILABLE FOR SUBSITUTION. ***')
 3000 CONTINUE
 4000 CONTINUE
 5000 CONTINUE
      RETURN
      END
```

89

3.2.16 PROGRAM CROPCALN

CROPCALN reads the weather data and models the crop progress.

3.2.16.1 Linkages

CROPCALN calls subroutines INIT, SWAP, START, PHENO and TERM.

3.2.16.2 Interfaces

CROPCALN is run after file CROPDATA has been edited.

3.2.16.3 Inputs

The old master file is assumed on Unit 1, INDEX on Unit 2, and CROPDATA on Unit 4.

3.2.16.4 Outputs

Unit 6 is the printer, GRIDINFO is on Unit 8, the new master file is on Unit 9, and ARNO is on Unit 11.

3.2.16.5 Description

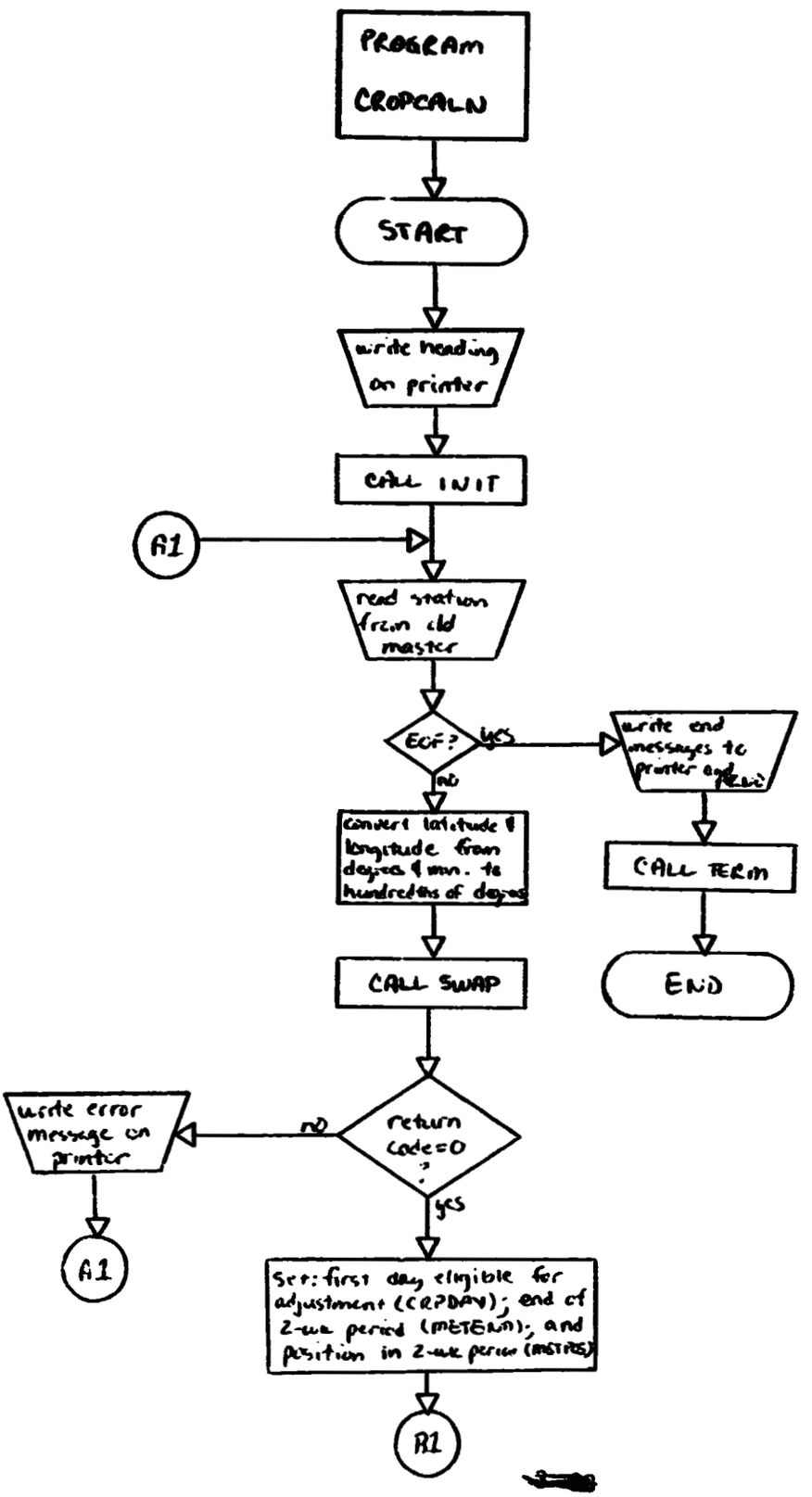
CROPCALN begins by writing a header to the printer and calling INIT. The program then enters a loop and reads a station. SWAP is called to obtain weather data from CROPDATA; CROPCALN checks and adjusts for a two-week period that spans two years and verifies the data obtained from CROPDATA, writing error messages ceasing operations for that particular station when errors are encountered. If the crop stage has not yet reached 1.0, START is called. If STAGE is between 1.0 and 6.0, CROPCALN goes into an inner loop; the daily crop progress for the two-week period is calculated via the subroutine PHENO and written to file GRIDINFO. Information is also written to file ARNO and the new master file. After all stations have been processed, ending messages are written to ARNO, the new master file, and the printer. TERM is called to print messages and close CROPDATA.

3.2.16.6 Flowchart

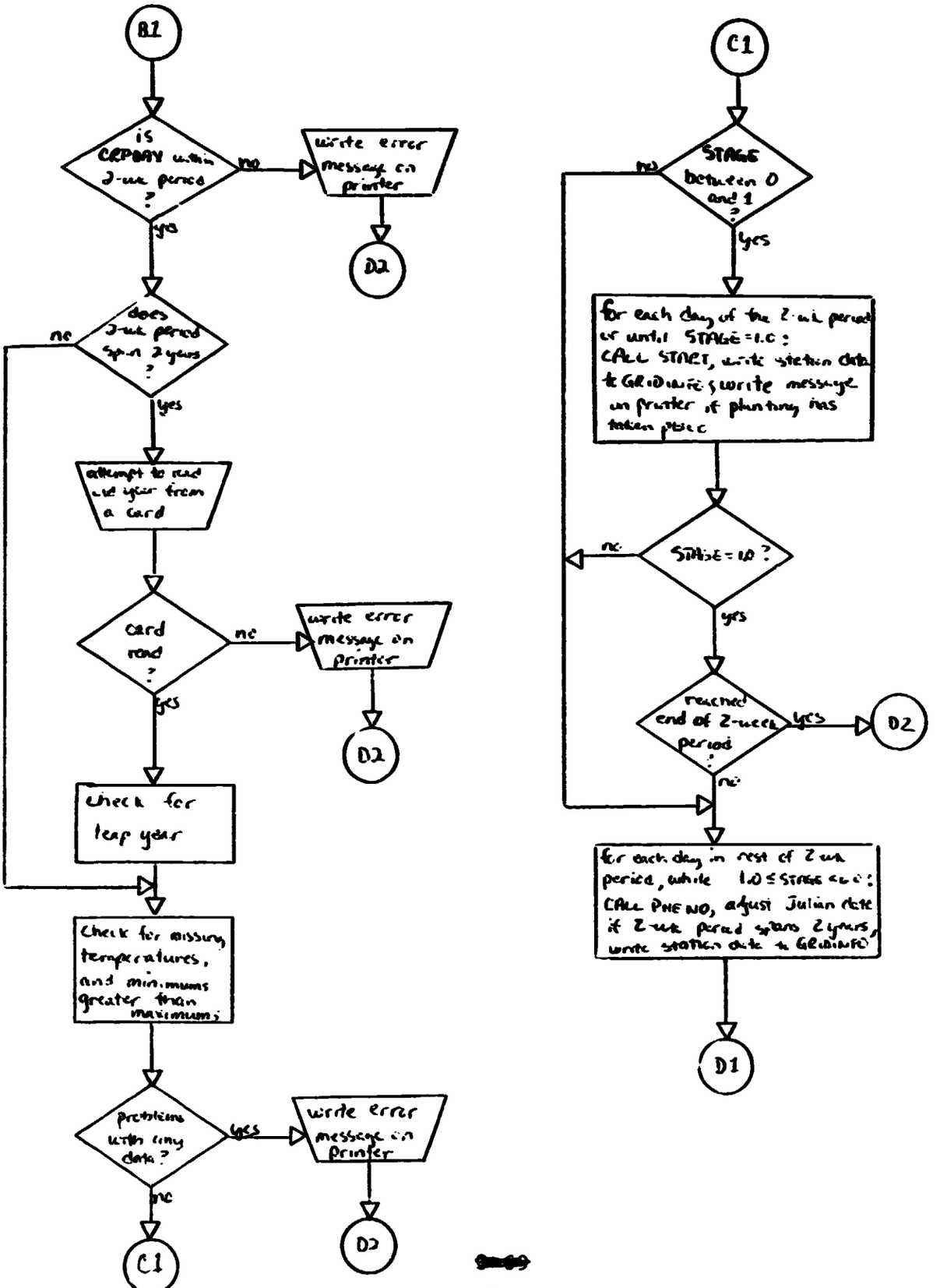
Next page.

3.2.16.7 Listing

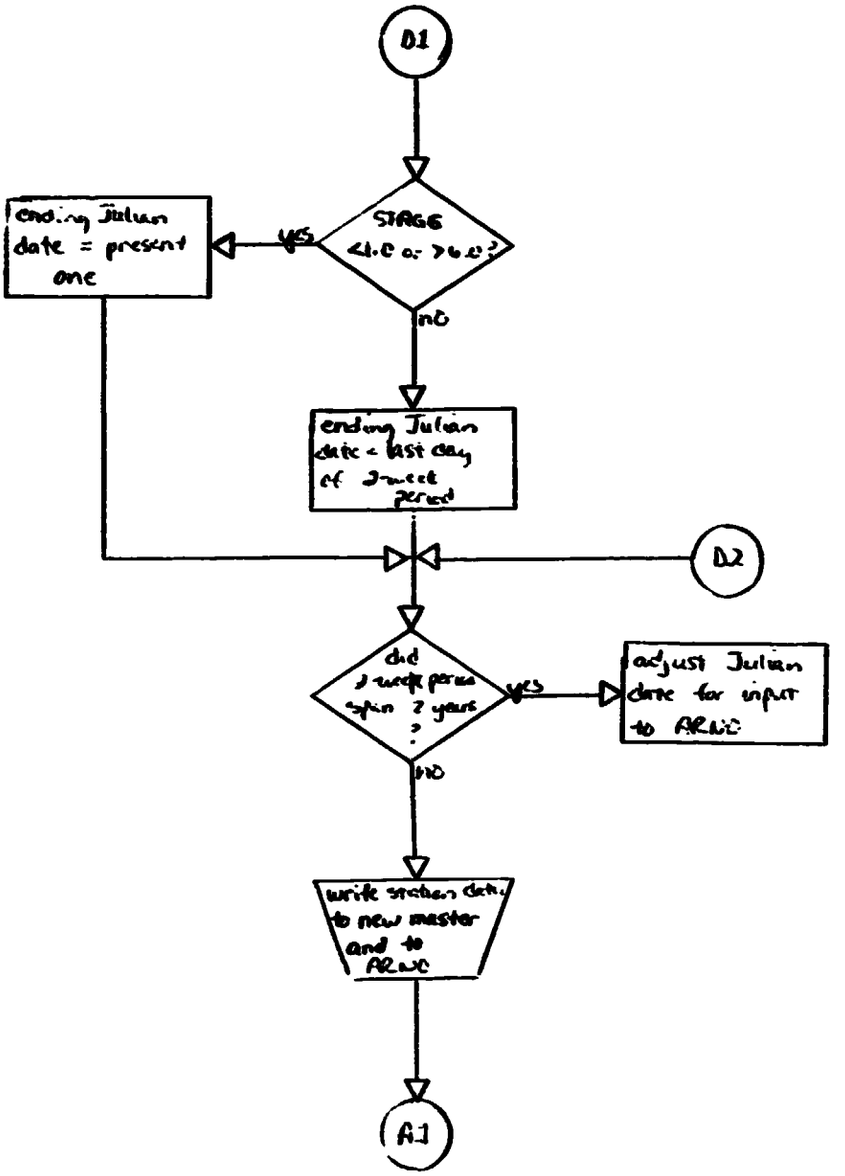
Follows flowchart.


PROGRAM CROPCALN
START
write heading on printer
CALL INIT
A1
read station from old master
EOF?
yes
no
write end messages to printer and close
CALL TERM
END
convert latitude & longitude from degrees & min. to hundredths of degrees
CALL SWAP
return code=0 ?
no
yes
write error message on printer
A1
Set: first day eligible for adjustment (CRPDAY); end of 2-wk period (METERM); and position in 2-wk period (METPOS)
A1

PROGRAM CROPCALN, CONST



PROGRAM CLIPCRIA, CRAFT



PROGRAM CROPCALN

```
      REAL MULT(5),STAGE,ALAT,ALONG,CDEV
      INTEGER [illegible],COU,REGION,ZONE,STRATA,LAT,LONG,CROP,JULIAN
      INTEGER [illegible],TEMPS(14,2),METBEG,METEND,CRRDAY,TODAY,PAGE/0/
      INTEGER [illegible]
      WRITE([illegible])
  2   FORMAT('1CROP CALENDAR---PHASE 3---CCEA/COLUMBIA, MO---ERROR MESSA
     *GES',/,[illegible])
      CALL INIT
      L=0
  1   READ(1,5,END=300)[illegible],COU,REGION,ZONE,STRATA,LAT,LONG,CDEV,CROP,
     * JULIAN,STAGE,MULT
  5   FORMAT(I5,[illegible],I6,I7,F5.2,I3,I4,F7.4,5F5.2,2I3)
      ALAT = (LAT/100)+FLOAT(MOD(LAT,100))/60.0
      ALONG=(LONG/100)+FLOAT(MOD(LONG,100))/60.0
C
C  GET MET DATA
C
      CALL SEAR(IN,[illegible],TEMPS,METBEG,IER)
      IF (IER) 220 , 10 , 220
C
C  CHECK DAY RANGE TO SEE IF ANY DATA TO PROCESS
C
 10   CRRDAY = JULIAN + 1
      METEND = METBEG + 13
      METPOS = CRRDAY + 1 - METBEG
      IF (METEND.LT.CRRDAY)  GO TO 106
      IF (METBEG.GT.CRRDAY)  GO TO 103
C
C CHECK IF PERIOD SPANS TWO YEARS; IF SO,READ IN FIRST YEAR
C
      IF(METEND.LE.365 .OR. L.EQ.1)GO TO 15
      L=1
      READ(5,11,END=104)YEAR
  11  FORMAT(F4.0)
      K=0
      CHECK=YEAR/4
      ICHECK=CHECK
      IF (ICHECK.EQ.CHECK)K=1
  15  CONTINUE
C
C  CHECK MET DATA
C
      DO 30 I = 1 , 14
      DO 20 J = 1 , 2
      IF (TEMPS(I,J).EQ.9999)  GO TO 110
 20   CONTINUE
      IF (TEMPS(I,1).GE.TEMPS(I,2))  GO TO 101
 30   CONTINUE
C
C  RUN STARTER MODEL, IF NEEDED
C
      DO 40 TODAY = CRRDAY , METEND
      IF ([illegible]) GO TO 50
      CALL STRT(CDEV,STAGE,CROP,TEMPS(METPOS,1),TEMPS(METPOS,2))
      WRITE([illegible])[illegible],COU,REGION,ZONE,STRATA,LAT,LONG,CDEV,CROP,TODAY,
     * STAGE,MULT,TEMPS(METPOS,1),TEMPS(METPOS,2)
      IF(CDEV.LE.[illegible])GOTO 40
      WRITE(6,35)[illegible],TODAY
  35  FORMAT(' STATION ',I5,' PLANTING 50% COMPLETE ON',I4)
 40   METPOS = METPOS + 1
      TODAY = METEND
      JULIAN = METEND
      IF(STAGE.GE.1.0)GOTO 200
```

```
C
C  RUN PHENOLOGICAL MODEL, IF APPROPRIATE
C
 50    NOW = TODAY
       IF (TODAY.GT.METEND)  GO TO 200
       DO 60 TODAY = NOW , METEND
       IF ((STAGE.LT.1.0).OR.(STAGE.GE.6.0))  GO TO 70
       CALL PHENO(TODAY,STAGE,CROP,TEMPS(METPOS,1),TEMPS(METPOS,2),MULT,
     * ALAT,ALONG)
       ITODAY=TODAY
       IF(ITODAY.GT.365)ITODAY=ITODAY-(365*K)
       IF(ITODAY.EQ.366 .AND. K.EQ.0)ITODAY=1
       WRITE(6,5)WMO,COU,REGION,ZONE,STRATA,LAT,LONG,CDEV,CROP,ITODAY,
     * STAGE,MULT,TEMPS(METPOS,1),TEMPS(METPOS,2)
 60    METPOS = METPOS + 1
       JULIAN = METEND
       GO TO 200
 70    JULIAN = TODAY
       GO TO 200
C
C  ERROR ROUTINES
C
 100   WRITE(6,1001)I,J,WMO
 1001  FORMAT(' MISSING DATA: ',2I4,' STATION',I6)
       GO TO 200
C
 101   WRITE(6,1011)I,J,WMO
 1011  FORMAT(' INVERTED TEMPS ',2I4,' STATION',I6)
       GO TO 200
C
 102   WRITE(6,1021)METEND,CURDAY,WMO
 1021  FORMAT(' MET DATA FOR PAST PERIOD: ',2I4 ,' STATION',I6)
       GO TO 200
C
 103   WRITE(6,1031)CURDAY,METEND,WMO
 1031  FORMAT(' MET DATA FOR FUTURE PERIOD: ',2I4,' STATION',I6)
       GO TO 200
C
  104  WRITE(6,1041)
 1041  FORMAT(' ERROR: TWO YEARS SPANNED: FIRST YEAR MUST BE INPUT ')
C
  200  IJUL=JULIAN
       IF(IJUL.GT.366)IJUL=IJUL-(365*K)
       IF(IJUL.EQ.366 .AND. K.EQ.0)IJUL=1
       WRITE(6,5)WMO,COU,REGION,ZONE,STRATA,LAT,LONG,CDEV,CROP,IJUL,
     *  STAGE,MULT
       LINE = LINE + 1
       IF (LINE.LE.50)  GO TO 210
       PAGE = PAGE + 1
       WRITE(11,2001) PAGE
 2001  FORMAT('1',24X,'CROP CALENDAR SUMMARY PRINTOUT',15X,'PAGE',I4,/,'0
     *STATION----CO-RG-ZO-ST----LATITUDE---LONGITUDE-DEVELOPMNT-CROP--J.
     * DAY----STAGE',/)
       LINE = 1
 210   WRITE(11,2101)WMO,COU,REGION,ZONE,STRATA,ALAT,ALONG,CDEV,CROP,
     * IJUL,STAGE
 2101  FORMAT(2I7,3I3,2F12.2,F9.2,I6,I8,F10.3)
       GO TO 1
C
 220   WRITE(6,2201)WMO
 2201  FORMAT(' STATION',I6,' NOT FOUND.  IGNORED.')
       GO TO 1
C
 300   WRITE(11,3001)
 3001  FORMAT('1END CROP CALENDAR SUMMARY')
       WRITE(6,3002)
 3002  FORMAT('-END OF REPORT')
       CALL TERM
```

75

3.2.17 JCL FILE SUZYQ

Executing JCL file SUZYQ invokes the crop calendar model.

3.2.17.1 Linkages

None.

3.2.17.2 Interfaces

None.

3.2.17.3 Inputs

None.

3.2.17.4 Outputs

None.

3.2.17.5 Description

JCL file SUZYQ binds the files referenced by CROPCALN to that program and request the invocation of CROPCALN.

3.2.17.6 Listing

```
//SUZYQØPROCØOLD=,NEW=,DSP=MOD
//TRACKØEXECØPGM=CROPCALN
//STEPLIBØDDØDSN=W.EDS.CCEA.PHASE3.LOAD,DISP=SHR
//FT01F001ØDDØDSN=W.EDS.CCEA.{GROUPI  }.MASTER&OLD.,DISP=SHR,
                             {GROUPII }
//ØØØDCB=BUFNO=1
//FT02F001ØDDØDSN=W.EDS.CCEA.{GROUPI  }.INDEX,DISP=SHR,DCB=BUFNO=1
                             {GROUPII }
//FT04F001ØDDØDSN=W.EDS.CCEA.{GROUPI  }.CROPDATA,DISP=SHR,
                             {GROUPII }
//ØØØDCB=BUFNO=1
//FT06F001ØDDØSYSOUT=A
//FT08F001ØDDØDSN=W.EDS.CCEA.{GROUPI  }.GRIDINFO,DISP=&DSP.,
                             {GROUPII }
//ØØØDCB=BUFNO=1
//FT09F001ØDDØDSN=W.EDS.CCEA.{GROUPI  }.MASTER&NEW., DISP=SHR,
                             {GROUPII }
//ØØØLCB=BUFNO=1
//FT11F001ØDDØDSN=W.EDS.CCEA.{GROUPI  } .ARNO,DISP=SHR,DCB=BUFNO=1
                             {GROUPII }
```

```
//REPORTØEXECØPGM=IEBPTPCH
//SYSUT1ØDDØDSN=W.EDS.CCEA.{GROUPI  }.ARNO,DISP=SHR
                           {GROUPII }
//SYSINØDDØDSN=W.EDS.CCEA.DATALIB(PREFORM),DISP=SHR
//SYSPRINTØDDØDUMMY
//SYSUT2ØDDØSYSOUT=A
//ØØØØØPEND
```

3.2.18 SUBROUTINE START

START accumulates temperature data to determine the planting date for individual spring wheat crop calendar stations.

3.2.18.1 Linkages

None.

3.2.18.2 Interfaces

None.

3.2.18.3 Inputs and Outputs

CDEV	cumulative development variable
STAGE	value of Robertson Scale
CROP	winter wheat = 11, spring wheat = 12
MIN	minimum temperature for a given day
MAX	maximum temperature for a given day

3.2.18.4 Description

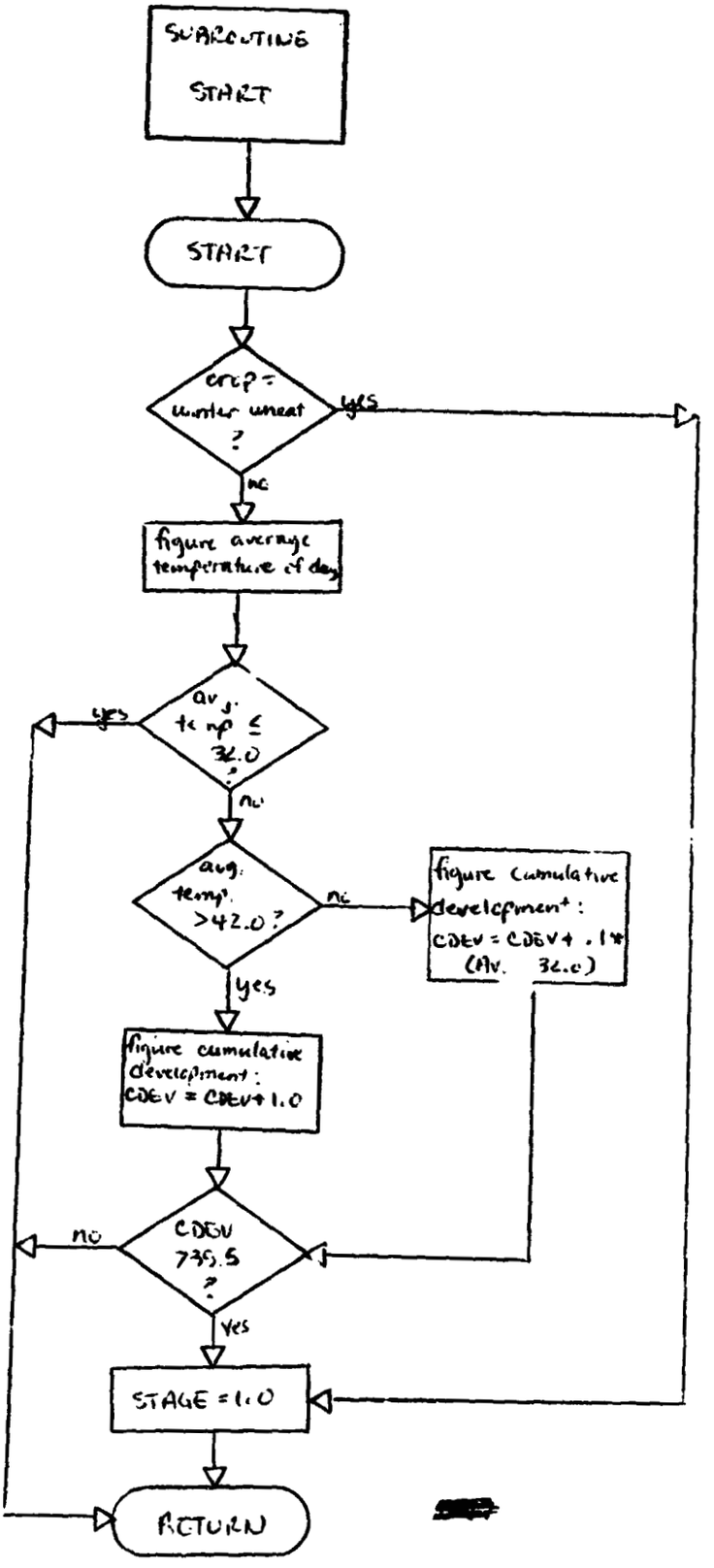
Subroutine START automatically sets STAGE to 1.0 for winter wheat stations. Spring wheat stations must reach a CDEV>35.5 before STAGE is set to 1.0. For average daily temperatures greater than 42°F, CDEV is increased by the value of 1.0.When average daily temperatures fall between 32°F and 42°F, CDEV is increased by one tenth the difference between the day's average and 32°F.

3.2.18.5 Flowchart

Next page.

3.2.18.6 Listing

Follows flowchart.


SUBROUTINE START
START
crop = winter wheat ?
yes
no
figure average temperature of day
Av. temp ≤ 32.0 ?
yes
no
avg. temp. >42.0 ?
no
figure cumulative development: CDEV = CDEV + .1* (Av. 32.0)
yes
figure cumulative development: CDEV = CDEV + 1.0
CDEV 735.5 ?
no
yes
STAGE = 1.0
RETURN

```
      SUBROUTINE START(CDEV,STAGE,CROP,MIN,MAX)
      INTEGER CROP,MIN,MAX
      REAL CDEV,STAGE,AVG
      IF(CROP.EQ.11)GOTO 100
      AVG=FLOAT(MIN+MAX)/2.0
      IF(AVG.LE.32.0)GOTO 200
      IF(AVG.GT.42.0)GOTO 3
      CDEV=CDEV+0.1*(AVG-32.0)
      GOTO 4
    3 CDEV=CDEV+1.0
    4 IF(CDEV.LE.35.5)GOTO 200
  100 STAGE=1.0
  200 RETURN
      END
```


80

3.2.19 SUBROUTINE PHENO

PHFNO is the routine which converts max and min temperatures and daylength into crop progress via Robertson's model with Feyerherm's modifications.

3.2.19.1 Linkages

PHENO calls real function DL.

3.2.19.2 Interfaces

None.

3.2.19.3 Inputs and Outputs

TODAY	Julian day being considered
STAGE	value of Robertson Scale
CROP	winter wheat = 11, spring wheat = 12
MIN	minimum temperature for a given day
MAX	maximum temperature for a given day
MULT	an array of 5 stage- and station-specific multipliers
ALAT	station latitude in hundreths of degrees
ALONG	station longitude in hundreths of degrees

3.2.19.4 Description

Subroutine PHENO calls function DL to compute the daylength of the day in question. The daylength, maximum and minimum temperatures are adjusted according to the stage of the previous day. Three stage advance factors are calculated using stage-specific coefficients: V1 uses daylength, V2 maximum temperature, and V3 minimum temperature. AINCR, the day's crop progress, is computed using the three stage advance factors and a station- and stage-specific multiplier; this is added to STAGE. If the crop stage is advancing too quickly during the fall, STAGE is held constant until a certain Julian date is reached. A check is made to see if the integer portion of STAGE has changed during the above calculations. If no change has occurred, operations return to the main program. When change in the integer portion of STAGE does occur, the amount of change (VT) is calculated, and the day's progress

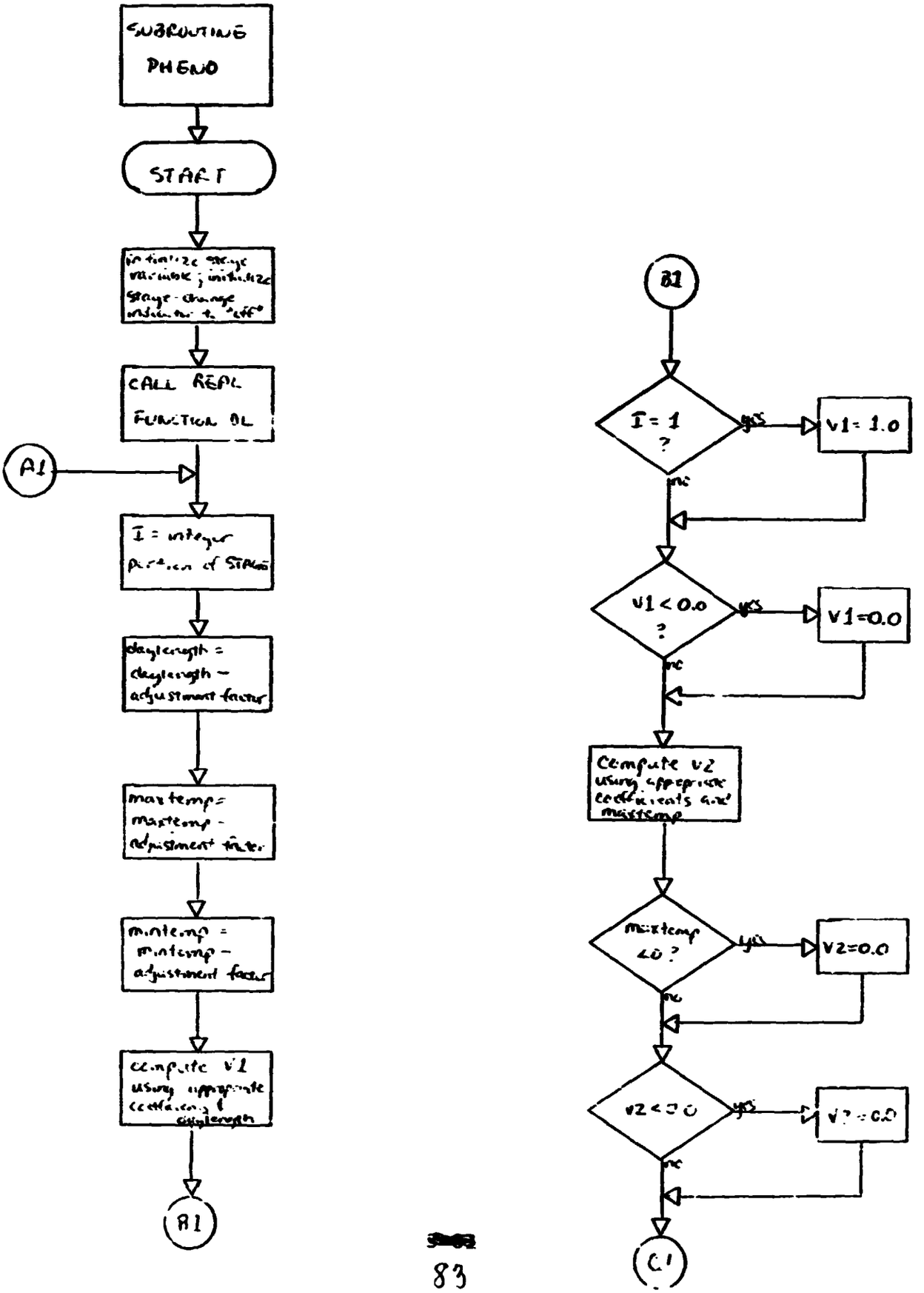
is refigured. The value of STAGE to be returned to the main program is then calculated, adding the integer portion of STAGE to the product of the PCNT and AINCR.

3.2.19.5 Flowchart

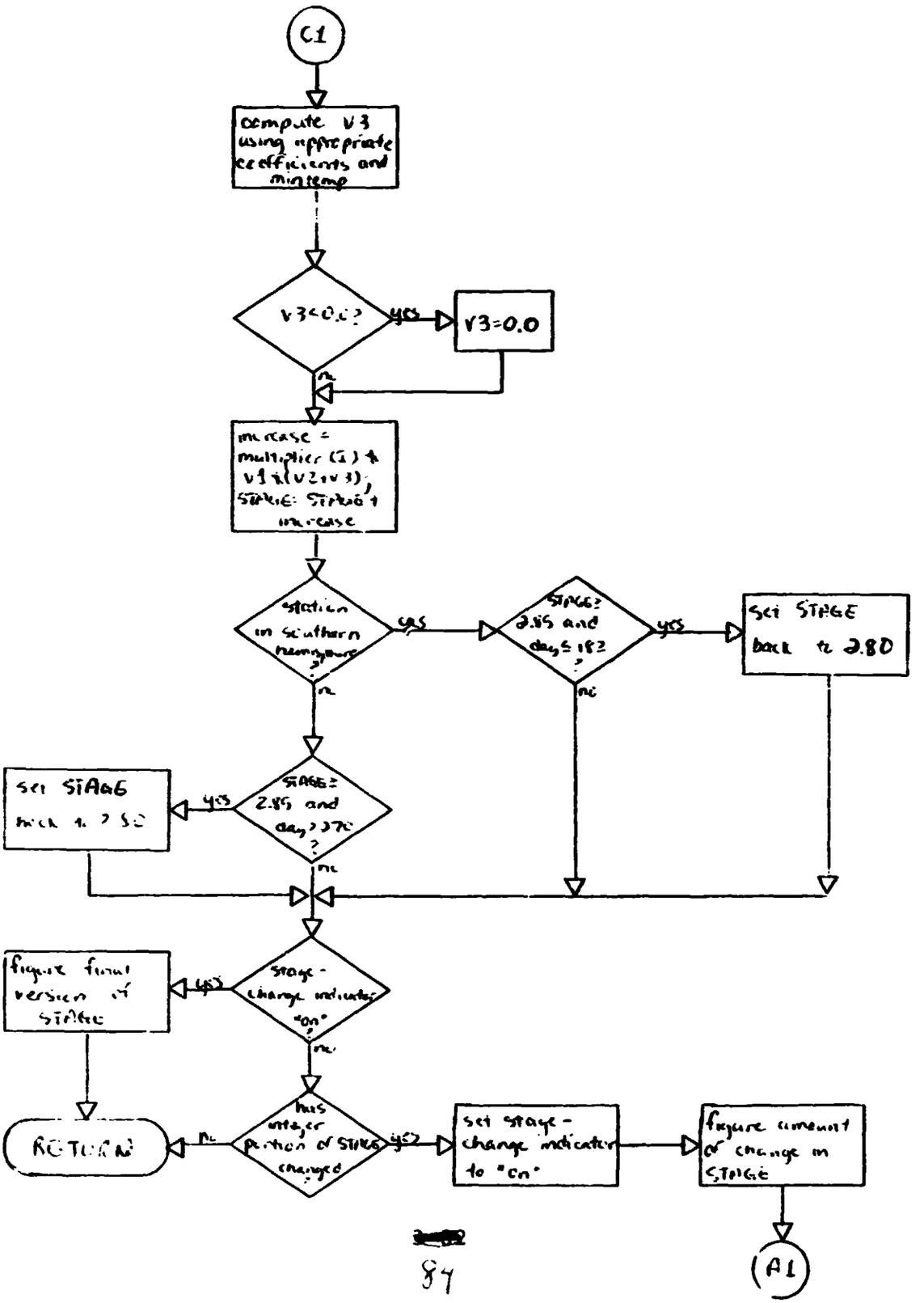
Next page.

3.2.19.6 Listing

Follows flowchart.


SUBROUTINE PHEND
START
CALL REPL FUNCTION BL
A1
I = integer
daylength = daylength - adjustment factor
maxtemp = maxtemp - adjustment factor
mintemp = mintemp - adjustment factor
compute V1
A1
B1
I = 1 ?
yes
V1 = 1.0
no
V1 < 0.0 ?
yes
V1=0.0
no
Compute V2
maxtemp < 0 ?
yes
V2=0.0
no
V2 < 0.0
yes
no
C1

SUBROUTINE PHENO, CONT



```
      SUBROUTINE PHENO(TODAY,STAGE,CROP,MIN,MAX,MULT,ALAT,ALONG)
      INTEGER TODAY,CROP,I,N,IA
      REAL STAGE,ALAT,ALONG,MULT(5),C(5,8),H,TX,TN,AINCR,V1,V2,V3
      DATA C(1,1)/40.37/,C(1,2)/.01169/,C(1,3)/-.000223/
      DATA C(1,4)/.0742/,C(1,5)/-.000259/,C(1,6)/-.134/
      DATA C(1,7)/1.665/,C(1,8)/0.0/,C(2,1)/3.49/,C(2,2)/-.00135/
      DATA C(2,3)/.0000425/,C(2,4)/.000305/,C(2,5)/-.0000042/
      DATA C(3,1)/10.33/,C(3,2)/.1259/,C(3,3)/-.000657/,C(3,4)/42.65/
      DATA C(3,5)/.0002459/,C(3,6)/0.0/,C(3,7)/.0003443/,C(3,8)/0.0/
      DATA C(4,1)/10.94/,C(4,2)/1.349/,C(4,3)/-.0419/,C(4,4)/42.18/
      DATA C(4,5)/.0002458/,C(4,6)/0.0/,C(4,7)/.0003109/,C(4,8)/0.0/
      DATA C(5,1)/24.38/,C(5,2)/-1.14/,C(5,3)/0.0/,C(5,4)/37.67/
      DATA C(5,5)/.0005733/,C(5,6)/0.0/,C(5,7)/.0003442/,C(5,8)/0.0/
      OLDSTG=STAGE
      JNT=0
      H=DL(ALAT,TODAY)
      TX=MAX
      TN=MIN
   10 I=STAGE
      H=H-C(I,1)
      TX=TX-C(I,4)
      TN=TN-C(I,4)
      V1=C(I,2)*H+C(I,3)*H*H
      IF(I.EQ.1) V1=1.0
      IF(V1.LT.0.0) V1=0.0
      V2=C(I,5)*TX+C(I,6)*TX*TX
      IF(TX.LT.0.0) V2 = 0.0
      IF(V2.LT.0.0) V2=0.0
      V3=C(I,7)*TN+C(I,8)*TN*TN
      IF(V3.LT.0.0) V3=0.0
      AINCR=MULT(I)*V1*(V2+V3)
      STAGE=STAGE+AINCR
      IF(ALAT.LT.0.0) GO TO 1
      IF(STAGE.GE.2.45.AND.TODAY.GT.270) STA  2.80
      GO TO 2
    1 IF(STAGE.GE.2.45.AND.TODAY.LE.182) ST   2.80
    2 IF(JNT.EQ.1) GO TO 11
      N=INT(OLDSTG)
      N=INT(STAGE)
      IF(N.EQ.N) GO TO 3
      JNT=1
      PCNT=(1.0-(N-OLDSTG)/AINCR)
      GO TO 10
   11 STAGE=N+PCNT*AINCR
    3 RETURN
      END
```

3.2.20 FUNCTION DL

DL converts latitude and date to daylength in hours.

3.2.20.1 Linkage

None.

3.2.20.2 Interfaces

None.

3.2.20.3 Inputs

XLAT	station latitude in hundreths of degrees
DATE	Julian day being considered

3.2.20.4 Outputs

Daylength in hours.

3.2.20.5 Description

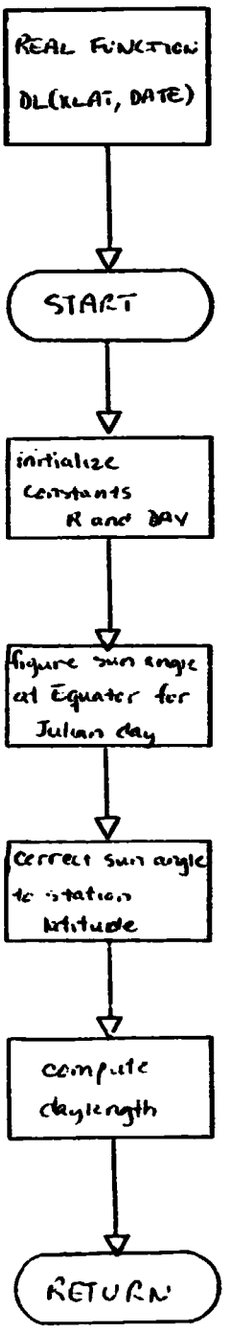
After initializing constants, the function calculates the angle of the sun on the equator for the date and then adjusts the angle according to the station's latitude. The daylength is computed using the product of the arc cosine of this angle and a constant value.

3.2.20.6 Flowchart

Next page.

3.2.20.7 Listing

Follows flowchart.


REAL FUNCTION
DL(XLAT, DATE)
START
initialize
constants
R and DAY
figure sun angle
at Equator for
Julian day
correct sun angle
to station
latitude
compute
daylength
RETURN

```
REAL FUNCTION DL(XLAT,DATE)
REAL XLAT,DAY,EPH,COH
INTEGER DATE
R=0.0174532925
DAY = DATE
EPH=23.5*SIN(0.9863*(DAY-80.0)*R)
COH = -TAN(XLAT*R)*TAN(EPH*R)
DL=ARCOS(COH)*7.6408787
RETURN
END
```

3.2.21 PROGRAM INSDEL

INSDEL is the main routine of the data replacement module. It allows the individual replacement of data in the weather file.

3.2.21.1 Linkages

Program INSDEL call subroutines INIT, SWAP and TERM.

3.2.21.2 Interfaces

INSDEL is run only when the EDIT procedure cannot satisfactorily edit the entire contents of CROPDATA. If needed, INSDEL is executed before program JSCTAPE can be run.

3.2.21.3 Inputs

The INDEX file is on Unit 2, CROPDATA on Unit 4, and the card input containing data corrections is on Unit 5.

3.2.21.4 Outputs

Corrections are made in CROPDATA. Some error messages may appear.

3.2.21.5 Description

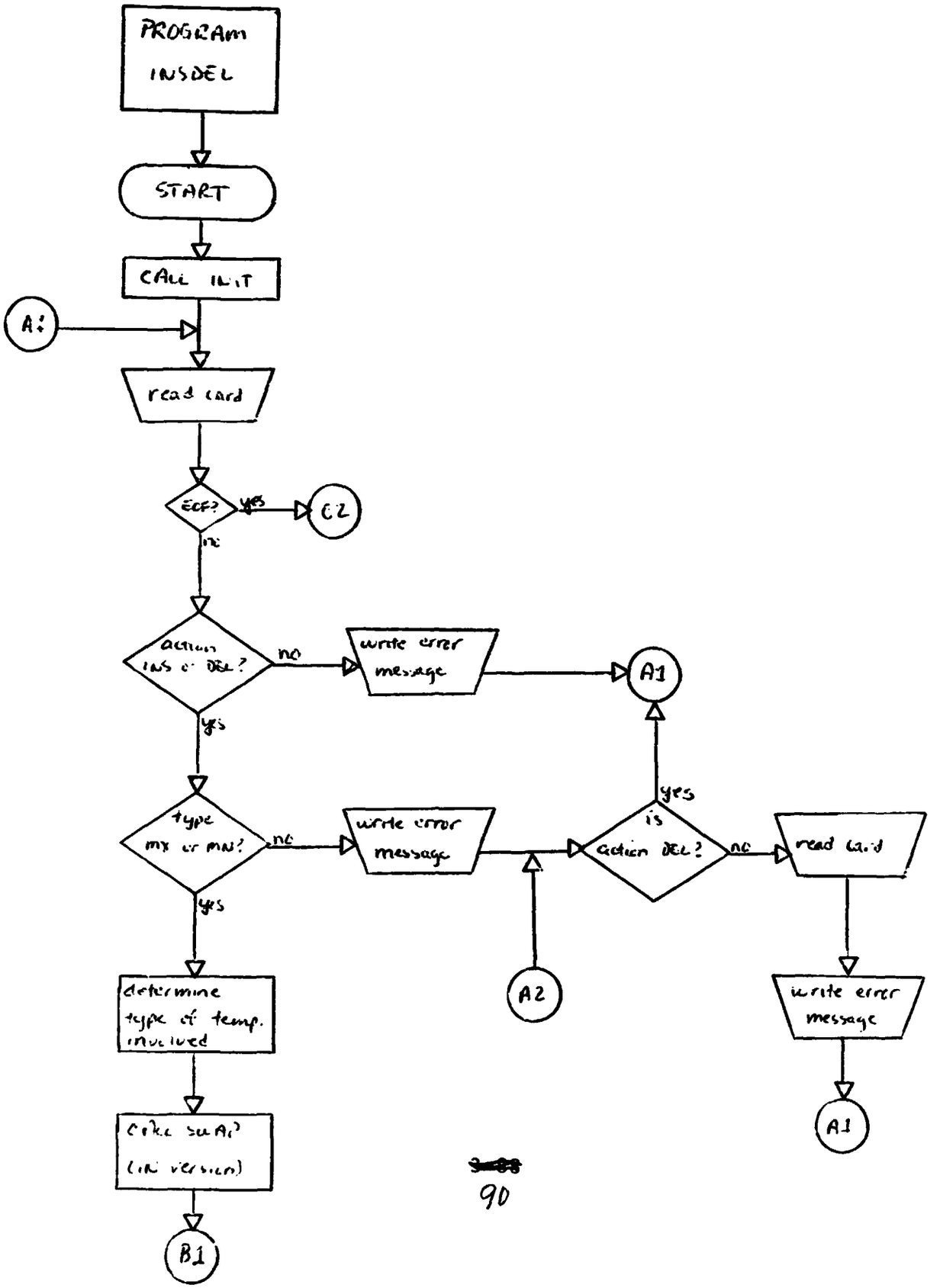
After calling INIT, INSDEL reads a card and scans it to see if it is an 'INS'ert or a 'DEL'ete card. Using SWAP, INSDEL then reads the span of days and type of temperature to be changed. If 'DEL' was indicated, the appropriate temperatures are set to missing (9999); if INS was called for, a card containing the new temperatures is read and then temperatures are inserted into CROPDATA. Any time problems are encountered with the card input, an error message is printed and the program proceeds to the next card. After all cards have been processed, TERM is called.

3.2.21.6 Flowchart

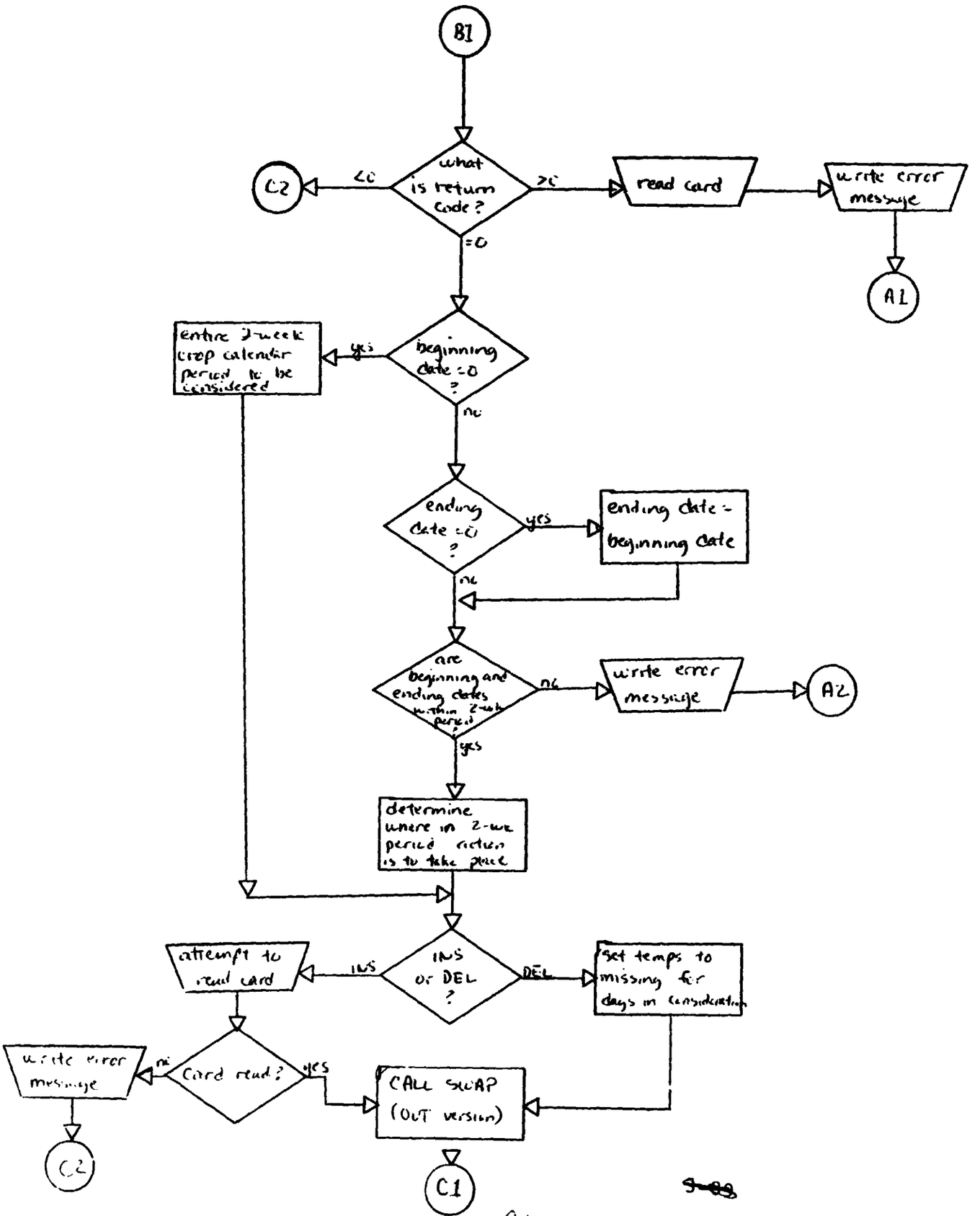
Next page.

3.2.21.7 Listing

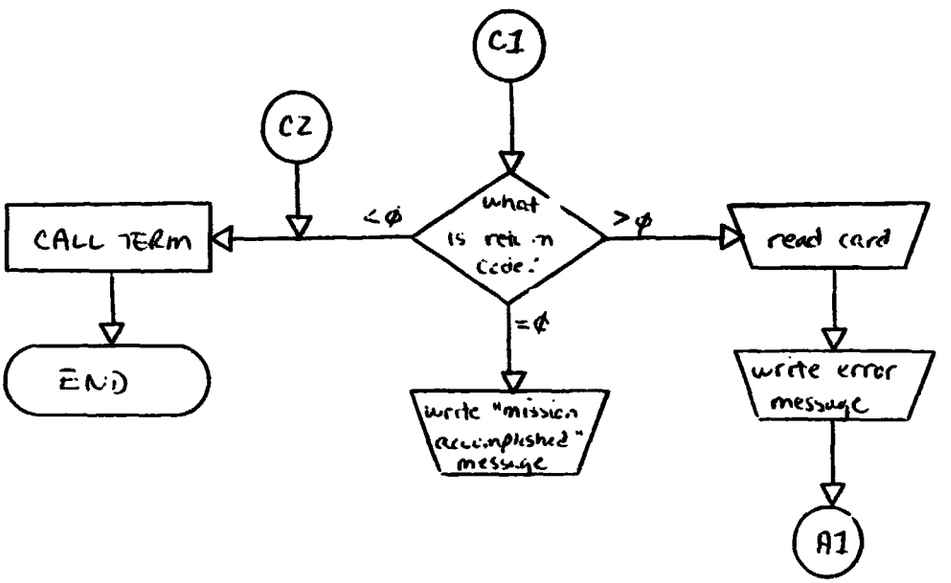
Follows flowchart.


PROGRAM INSDEL
START
CALL INIT
A1
read card
EOF?
yes
C2
no
action INS or DEL?
no
write error message
A1
yes
type mx or mN?
no
write error message
A2
is action DEL?
yes
no
read card
write error message
A1
yes
determine type of temp. involved
Take suA1? (IN version)
B1

PROGRAM INSDEL, CONT



PROGRAM INSDEL, CON'T



```
      INTEGER OUT/2/,IN/1/,FIRST,LAST,TEMPS(14,2),MISSNG/99999/,IREG
      INTEGER TRNO,ACTION,WORDNM,BEGIN,END,DEL/'DEL '/,INS/'INS '/
      INTEGER RCODE
      INTEGER*2 TYPE,TYPES(2)/'NM','BX'/
      CALL INIT
    1 READ(5,10,END=100)ACTION,WORDNM,TYPE,BEGIN,END
   10 FORMAT(A4,I2,1X,A2,2(1X,I3))
      RCODE=0
      IF ((ACTION.NE.DEL).AND.(ACTION.NE.INS))  GO TO 900
      IF ((TYPE.NE.TYPES(1)).AND.(TYPE.NE.TYPES(2))) GO TO 910
      ITYPE=1
      IF (TYPE.EQ.TYPES(2))ITYPE=2
      CALL SWAP(IN,WORDNM,TEMPS,FIRST,RCODE)
      IF (RCODE) 100,15,915
   15 IF (BEGIN.EQ.0)  GO TO 20
      IF (END.EQ.0)END=BEGIN
      LAST=FIRST+13
      IF ((BEGIN.LT.FIRST).OR.(BEGIN.GT.LAST))  GO TO 920
      IF ((END.LT.FIRST).OR.(END.GT.LAST))  GO TO 930
      IBEG=BEGIN-FIRST+1
      IEND=END-FIRST+1
      GO TO 30
   20 IBEG=1
      IEND=14
      BEGIN=FIRST
      END=FIRST+13
   30 IF (ACTION.EQ.INS)  GO TO 50
      DO 40 I=IBEG,IEND
   40 TEMPS(I,ITYPE)=MISSNG
      GO TO 60
   50 READ(5,55,END=60)(TEMPS(I,ITYPE),I=IBEG,IEND)
   55 FORMAT(14I5)
   60 CALL SWAP(OUT,WORDNM,TEMPS,FIRST,RCODE)
      IF (RCODE) 100,65,915
   65 WRITE(6,66)ACTION,WORDNM,TYPE,BEGIN,END
   66 FORMAT('0',A4,'FOR STATION',I2,1X,A2,' TEMP(S) FROM',I4,' TO',I4,
     $' WAS SUCCESSFUL.')
      GO TO 1
  900 WRITE(6,901) ACTION
  901 FORMAT('0COMMAND ',A4,' NOT RECOGNIZED.  IT HAS BEEN IGNORED.')
      GO TO 1
  910 WRITE(6,911)TYPE,ACTION
  911 FORMAT('0DATA TYPE OF ',A2,' NOT RECOGNIZED. ',A4,
     $' COMMAND IGNORED.')
  915 IF (ACTION.EQ.DEL)  GO TO 1
      READ(5,10,END=100)
      WRITE(6,912)
  912 FORMAT(' 1 CARD FLUSHED DUE TO INS COMMAND.')
      GO TO 1
  916 READ(5,10,END=100)
      WRITE(6,917)
  917 FORMAT(' 1 CARD FLUSHED DUE TO END NOT FOUND.')
      GO TO 1
  920 WRITE(6,921)BEGIN,FIRST,LAST,ACTION
  921 FORMAT('0BEGIN DATE OF',I4,' NOT IN RANGE OF',I4,' TO',I4,
     $'.  ',A4,' COMMAND IGNORED.')
      GO TO 915
  930 WRITE(6,931)END,FIRST,LAST,ACTION
  931 FORMAT('0END DATE OF',I4,' NOT IN RANGE OF',I4,' TO',I4,
     $'.  ',A4,' COMMAND IGNORED.')
      GO TO 915
  940 WRITE(6,941)
  941 FORMAT('0END-OF-FILE ENCOUNTERED WHEN EXPECTING DATA FOR INS COMMAND.')
```

```
     5 ,' IOS COMMAND IGNORED.')
100 REWIND 5
     CALL TERM
     STOP
     END
```

ORIGINAL PAGE IS OF POOR QUALITY

94

3.2.22 JCL FILE INSERT

Executing JCL file INSERT invokes the INSDEL program.

3.2.22.1 Linkages

None.

3.2.22.2 Interfaces

None.

3.2.22.3 Inputs

Cards containing the information necessary to correct CROPDATA must be inserted. See Section 4.2.2.5 for formats.

3.2.22.4 Outputs

None.

3.2.22.5 Description

JCL file INSERT binds the files referenced by INSDEL to that program and requests the execution of INSDEL.

3.2.22.6 Listing

```
//INSERT  PROC
//INS  EXEC  PGM=INSDEL
//STEPLIB  DD  DSN=W.EDS.CCEA.PHASE3.LOAD,DISP=SHR
//FT02F001  DD  DSN=W.EDS.CCEA.{GROUPI  }.INDEX,DISP=SHR
                               {GROUPII }
//FT04F001  DD  DSN=W.EDS.CCEA.{GROUPI  }.CROPDATA,DISP=SHR
                               {GROUPII }
//FT05F001  DD  DDNAME=SYSIN
//FT06F001  DD  SYSOUT=A
//      PEND
```

3.2.23 PROGRAM DRECOVER

This program fills the INDEX file from card input.

3.2.23.1 Linkages

None.

3.2.23.2 Interfaces

Whenever changes are to be made in the INDEX file, DRECOVER is run to reconstruct the entire file.

3.2.23.3 Inputs

Input to DRECOVER is by cards; the first card contains the number of stations to be in the INDEX file, punched in columns 1-3. Each subsequent card contains a station to be in the file, and its three closest neighbor stations, in 4(I5,1X) format.

3.2.23.4 Outputs

The INDEX file is created.

3.2.23.5 Description

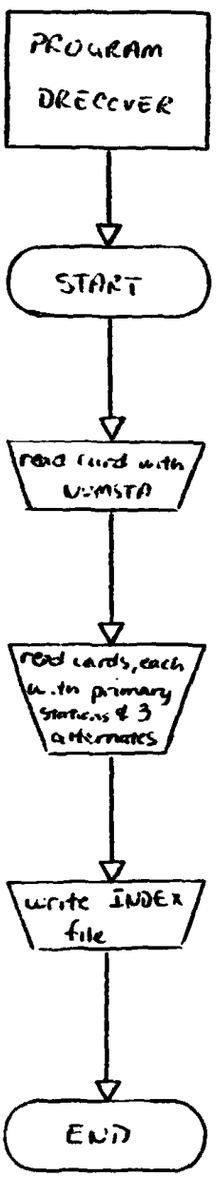
DRECOVER reads all input cards, sorting the data into NUMSTA (number of stations) and two arrays: WMO, containing all stations on the INDEX file, and STAIN, containing the three closest stations to each station in array, WMO. DRECOVER then writes all data to the INDEX file.

3.2.23.6 Flowchart

Next page.

3.2.23.7 Listing

Follows flowchart.


PROGRAM
DRECEVER
START
read card with
W.MSTA
read cards, each
w. its primary
station & 3
alternates
write INDEX
file
END

```
//DRECOVER JOB ('DD10008E1HEA  ','COLUM'),VONHOLT,REGION=256K,TIME=1
//SF EXEC NCDGOCGO
//SYSIN DD *
      INTEGER WMO(511),STATN(511,3)
      INTEGER*2 NUMSTA
      READ(5,1,END=2)NUMSTA,(WMO(I),(STATN(I,J),J=1,3),I=1,511)
    1 FORMAT(I3,/,255(4(I5,1X),/),255(4(I5,1X),/))
    2 WRITE(8,3)NUMSTA,(WMO(I),(STATN(I,J),J=1,3),I=1,511)
    3 FORMAT(A2,2X,255A4,/,10(2(128A4),/))
      STOP
      END
/*
//SYSGO DD *
/*
```

86

3.2.24 PROGRAM JSCTAPE

JSCTAPE is a PL/I program transferring data from disk file GRIDINFO to a tape to be sent to Houston.

3.2.24.1 Linkages

None.

3.2.24.2 Interfaces

JSCTAPE should not be executed until an error-free fun of CROPCALN has been processed.

3.2.24.3 Inputs

Disk file GRIDINFO and a card containing header information for the tape. See Section 4.2.3 for card forma'

3.2.24.4 Outputs

JSCTAPE outputs a tape copy of file GRIDINFO preceded by a header indicating the countries and Julian dates involved.

3.2.24.5 Description

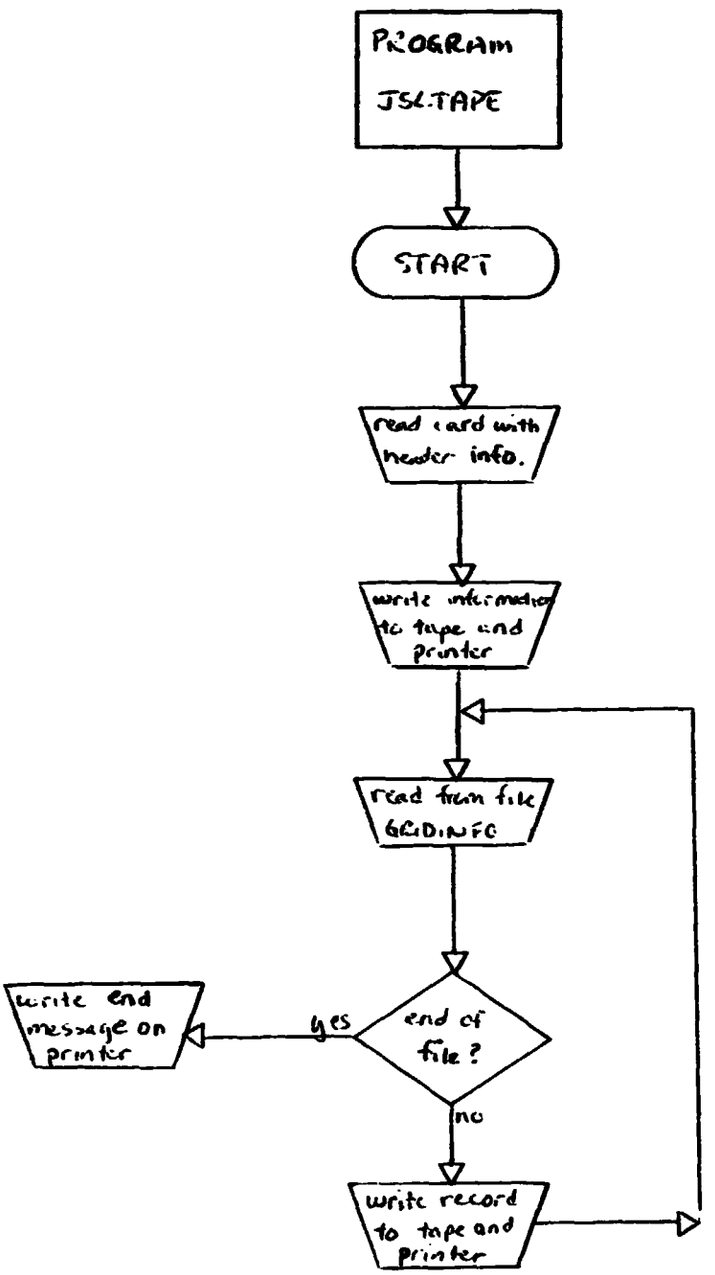
The program reads the card with header information and copies this information both to tape and to the printer. JSCTAPE proceeds to read each record of GRIDINFO and write the data to both the tape and the printer. When end-of-file is encountered, a closing message is written to the printer and operation ceases.

3.2.24.6 Flowchart

Next page.

3.2.24.7 Listing

Follows flowchart.


PROGRAM
JSLTAPE
START
read card with header info.
write information to tape and printer
read from file GRIDINFO
end of file?
yes
write end message on printer
no
write record to tape and printer

```
JSCTAPE: PROC OPTIONS(MAIN);
   DCL INSTR CHAR(80);
   DCL SYSIN FILE STREAM INPUT;
   DCL SYSPRINT FILE PRINT OUTPUT;
   DCL IN FILE RECORD INPUT;
   DCL OUT FILE RECORD OUTPUT;
   ON ENDFILE(IN) GOTO EXIT;
   OPEN FILE(SYSIN), FILE(SYSPRINT), FILE(IN), FILE(OUT);
   INSTR=' ';
   GET FILE(SYSIN) EDIT(INSTR)(COL(1),A(80));
   WRITE FILE(OUT) FROM(INSTR);
   PUT FILE(SYSPRINT) EDIT(INSTR)(SKIP,A(80));
 L100: READ FILE(IN) INTO(INSTR);
   WRITE FILE(OUT) FROM(INSTR);
   PUT FILE(SYSPRINT) EDIT(INSTR)(SKIP,A(80));
   GOTO L100;
 EXIT: PUT FILE(SYSPRINT) EDIT('*** END OF FILE IN ***')(SKIP(3),A);
   CLOSE FILE(SYSIN), FILE(SYSPRINT), FILE(IN), FILE(OUT);
 END JSCTAPE;
```

4. OPERATION

Operation of the Robertson Phenological Model is a mixed human/machine procedure. This document describes the mechanical procedures.

4.1 USER DOCUMENTATION

Most of the work involved in operating the crop calendar is in the preparation of meteorological (met) data. To do this properly requires a knowledge of the organization of the crop calendar and met data files.

Crop calendar countries are separated into two sets: Group I - the U.S., China, and India, and Group II - Canada and the U.S.S.R. Each set is run every two weeks with any given 14-day period ending on a Sunday, and the groups are run in alternating weeks (Group I one week, Group II the next). Therefore the 14-day periods comprising each crop calendar run for each group are not identical. For example, the 2-week periods in Julian days for Group I may be 52-65, 66-79, and 80-93, while the 14-day periods for Group II may be 59-72, 73-86, and 87-100.

Met data input is divided by months and by area (North America and non-North America). Previous months' data are on tape, while the current month's data are on disk. For each group there is a maximum of four met data collection runs per two-week period. (last month, this month) x (North America, non-North America).

4.2 PROCEDURES

In total, eight procedures are involved: one to construct the INDEX file, six to prepare data and run the Robertson Phenological Model, and one to prepare a tape of crop calendar data for use in Houston.

4.2.1 CONSTRUCTING THE INDEX FILE (DRECOVER)

Whenever changes are to be made in the INDEX file, DRECOVER is run to reconstruct the entire file. Card input follows the program cards as in the example below:

```
//FT08F001ØDDØDSN=W.EDS.CCEA.{GROUPI }.INDEX,DISP=SHR
                             {GROUPII}
//SYSGOØDDØ*
235
42809Ø42971Ø42798Ø42492
43003Ø43192Ø43014Ø43117
etc.
/*
//
```

4.2.2 CROP CALENDAR PROCEDURES

There are two sets of Job Control Language (JCL) PROC's; one for the Group I countries and one for Group II. The JCL accesses the partitioned dataset W.EDS.CCEA.PHASE3.LOAD containing all crop calendar programs in executable form. The deck for the appropriate group is input each time any crop calendar program is run. The EXEC card (or cards) for each program in the following sections of explanation refers to a PROC contained in these JCL decks, and is placed after the final //ØØØØØPEND card in the JCL deck.

The JCL decks for Group I and Group II are found on the following two pages.

JCL PROCEDURES FOR

GROUP I

```
//LIST PROC
//LIST EXEC PGM=PRINT
//STEPLIB DD DSN=W.EDS.CCEA.PHASE3.LOAD,DISP=SHR
//FT02F001 DD DSN=W.EDS.CCEA.GROUPI.INDEX,DISP=SHR
//FT04F001 DD DSN=W.EDS.CCEA.GROUPI.CROPDATA,DISP=SHR
//FT06F001 DD SYSOUT=A
//       PEND
//EDIT PROC
//EDIT EXEC PGM=EDITOR,TIME=(,15)
//STEPLIB DD DSN=W.EDS.CCEA.PHASE3.LOAD,DISP=SHR
//FT02F001 DD DSN=W.EDS.CCEA.GROUPI.INDEX,DISP=SHR
//FT04F001 DD DSN=W.EDS.CCEA.GROUPI.CROPDATA,DISP=SHR
//FT05F001 DD DUMMY
//FT06F001 DD SYSOUT=A
//       PEND
//SUZYQ PROC OLD=,NEW=,DSP=MOD
//TRACK EXEC PGM=CROPCALN
//STEPLIB DD DSN=W.EDS.CCEA.PHASE3.LOAD,DISP=SHR
//FT01F001 DD DSN=W.EDS.CCEA.GROUPI.MASTER&OLD.,DISP=SHR,DCB=BUFNO=1
//FT02F001 DD DSN=W.EDS.CCEA.GROUPI.INDEX,DISP=SHR,DCB=BUFNO=1
//FT04F001 DD DSN=W.EDS.CCEA.GROUPI.CROPDATA,DISP=SHR,DCB=BUFNO=1
//FT06F001 DD SYSOUT=A
//FT08F001 DD DSN=W.EDS.CCEA.GROUPI.GRIDINFO,DISP=&DSP.,DCB=BUFNO=1
//FT09F001 DD DSN=W.EDS.CCEA.GROUPI.MASTER&NEW.,DISP=SHR,DCB=BUFNO=1
//FT11F001 DD DSN=W.EDS.CCEA.GROUPI.ARNO,DISP=SHR,DCB=BUFNO=1
//REPORT EXEC PGM=IEBPTPCH
//SYSUT1 DD DSN=W.EDS.CCEA.GROUPI.ARNO,DISP=SHR
//SYSIN DD DSN=W.EDS.CCEA.DATALIB(PREFORM),DISP=SHR
//SYSPRINT DD DUMMY
//SYSUT2 DD SYSOUT=A
//       PEND
//CLEAR PROC
//CLEAR EXEC PGM=CLEAR
//STEPLIB DD DSN=W.EDS.CCEA.PHASE3.LOAD,DISP=SHR
//FT02F001 DD DSN=W.EDS.CCEA.GROUPI.INDEX,DISP=SHR
//FT04F001 DD DSN=W.EDS.CCEA.GROUPI.CROPDATA,DISP=SHR
//FT06F001 DD SYSOUT=A
//FT05F001 DD DDNAME=SYSIN
//       PEND
//INSERT PROC
//INS EXEC PGM=INSDEL
//STEPLIB DD DSN=W.EDS.CCEA.PHASE3.LOAD,DISP=SHR
//FT02F001 DD DSN=W.EDS.CCEA.GROUPI.INDEX,DISP=SHR
//FT04F001 DD DSN=W.EDS.CCEA.GROUPI.CROPDATA,DISP=SHR
//FT05F001 DD DDNAME=SYSIN
//FT06F001 DD SYSOUT=A
//       PEND
//CAPTURE PROC UNIT=,VOL=,DSN='W.EDS.CCEA.DAYDATA'
//CAPTURE EXEC PGM=CAPTURE
//STEPLIB DD DSN=W.EDS.CCEA.PHASE3.LOAD,DISP=SHR
//FT01F001 DD SYSOUT=A,DCB=(LRECL=100,BLKSIZE=1500,RECFM=FB)
//FT02F001 DD DSN=W.EDS.CCEA.GROUPI.INDEX,DISP=SHR
//FT03F001 DD DSN=&DSN.,DISP=SHR,UNIT=&UNIT.,VOL=SER=&VOL,LABEL=(,,,IN)
//FT04F001 DD DSN=W.EDS.CCEA.GROUPI.CROPDATA,DISP=SHR
//FT06F001 DD SYSOUT=A
//       PEND
```

```
//LIST PROC
//LIST EXEC PGM=PRINT
//STEPLIB DD DSN=W.EDS.CCEA.PHASE3.LOAD,DISP=SHR
//FT02F001 DD DSN=W.EDS.CCEA.GROUPII.INDEX,DISP=SHR
//FT04F001 DD DSN=W.EDS.CCEA.GROUPII.CROPDATA,DISP=SHR
//FT06F001 DD SYSOUT=A
//       PEND
//EDIT PROC
//EDIT EXEC PGM=EDITOR,TIME=(,15)
//STEPLIB DD DSN=W.EDS.CCEA.PHASE3.LOAD,DISP=SHR
//FT02F001 DD DSN=W.EDS.CCEA.GROUPII.INDEX,DISP=SHR
//FT04F001 DD DSN=W.EDS.CCEA.GROUPII.CROPDATA,DISP=SHR
//FT06F001 DD SYSOUT=A
//       PEND
//SUZYQ PROC OLD=,NEW=,DSP=MOD
//TRACK EXEC PGM=CROPCALN
//STEPLIB DD DSN=W.EDS.CCEA.PHASE3.LOAD,DISP=SHR
//FT01F001 DD DSN=W.EDS.CCEA.GROUPII.MASTER&OLD.,DISP=SHR,DCB=BUFNO=1
//FT02F001 DD DSN=W.EDS.CCEA.GROUPII.INDEX,DISP=SHR,DCB=BUFNO=1
//FT04F001 DD DSN=W.EDS.CCEA.GROUPII.CROPDATA,DISP=SHR,DCB=BUFNO=1
//FT06F001 DD SYSOUT=A
//FT08F001 DD DSN=W.EDS.CCEA.GROUPII.GRIDINFO,DISP=&DSP.,DCB=BUFNO=1
//FT09F001 DD DSN=W.EDS.CCEA.GROUPII.MASTER&NEW.,DISP=SHR,DCB=BUFNO=1
//FT11F001 DD DSN=W.EDS.CCEA.GROUPII.ARNO,DISP=SHR,DCB=BUFNO=1
//REPORT EXEC PGM=IEBPTPCH
//SYSUT1 DD DSN=W.EDS.CCEA.GROUPII.APNO,DISP=SHR
//SYSIN DD DSN=W.EDS.CCEA.DATALIB(PREFORM),DISP=SHR
//SYSPRINT DD DUMMY
//SYSUT2 DD SYSOUT=A
//       PEND
//INSERT PROC
//INS EXEC PGM=INSDEL
//STEPLIB DD DSN=W.EDS.CCEA.PHASE3.LOAD,DISP=SHR
//FT02F001 DD DSN=W.EDS.CCEA.GROUPII.INDEX,DISP=SHR
//FT04F001 DD DSN=W.EDS.CCEA.GROUPII.CROPDATA,DISP=SHR
//FT05F001 DD DDNAME=SYSIN
//FT06F001 DD SYSOUT=A
//       PEND
//CLEAR PROC
//CLEAR EXEC PGM=CLEAR
//STEPLIB DD DSN=W.EDS.CCEA.PHASE3.LOAD,DISP=SHR
//FT02F001 DD DSN=W.EDS.CCEA.GROUPII.INDEX,DISP=SHR
//FT04F001 DD DSN=W.EDS.CCEA.GROUPII.CROPDATA,DISP=SHR
//FT05F001 DD DDNAME=SYSIN
//FT06F001 DD SYSOUT=A
//       PEND
//CAPTURE PROC UNIT=,VOL=,DSN='W.EDS.CCEA.DAYDATA'
//CAPTURE EXEC PGM=CAPTURE
//STEPLIB DD DSN=W.EDS.CCEA.PHASE3.LOAD,DISP=SHR
//FT01F001 DD SYSOUT=A,DCB=(LRECL=100,BLKSIZE=1500,RECFM=FB)
//FT02F001 DD DSN=W.EDS.CCEA.GROUPII.INDEX,DISP=SHR
//FT03F001 DD DSN=&DSN.,DISP=SHR,UNIT=&UNIT.,VOL=SER=&VOL,LABEL=(,,IN
//FT04F001 DD DSN=W.EDS.CCEA.GROUPII.CROPDATA,DISP=SHR
//FT06F001 DD SYSOUT=A
//       PEND
```

4.2.2.1 INITIALIZING FILE CROPDATA (CLEAR)

The CLEAR program is called by the CLEAR procedure. This sets all temperatures in W.EDS.CCEA.{GROUPI / GROUPII}.CROPDATA to '9999'. The starting date for the crop calendar period being run is also determined by CLEAR. The default option will add 14 to the starting date of the last period to get the new starting date for the period being run. This may be overridden by user input, which is required if the previous two-week period spans two years. Examples are given below:

A. The default option.

```
//ØEXECØCLEAR
```

B. Setting the starting date 28 days after the last period.

```
//ØEXECØCLEAR
//FT05F001ØDDØ*
28
/*
```

C. Setting the starting date for a new year (in this example, the previous starting date is 362 and the next one is 010; the change is a -352 and the following input is used).

```
//ØEXECØCLEAR
//FT05F001ØDDØ*
-352
/*
```

4.2.2.2 FILLING FILE CROPDATA (CAPTURE)

The CAPTURE program is called by the CAPTURE procedure. This obtains maximum (MX) and minimum (MN) temperature data from catalogued disk files in the current month and from tape in past months. Four datasets can be accessed by CAPTURE:

1. W.EDS.CCEA.DAILY - to access North American data in the current month (disk file).

2. CCEA.xxx.SRTD - to access North American data in a past month (tape access) where 'xxx' is the month abbreviation (ex., CCEA.JAN.SRTD).

3. W.EDS.CCEA.DAYDATA - for non-North American data in the current month (disk file).

4. CCEA.DAILY.xxx - for non-North American data in a past month (tape access), where'xxx' is the month abbreviation (ex., CCEA.DAILY.JAN.).

Data for past months is put on tape at Suitland, Maryland. The appropriate dataset name (and unit and tape number, if needed) is input on the EXEC card. When accessing North American data (W.EDS.CCEA.DAILY or CCEA.xxx.SRTD) the Julian date of the first day of the month that is being run must be input. This is not needed for non-North American data (W.EDS. CCEA.DAYDATA or CCEA.DAILY.xxx). See examples below:

A. GROUP I (China, India, and the U.S.)

To CAPTURE data entirely within the current month, the following input is used:

```
//ØEXECØCAPTURE,DSN='W.EDS.CCEA.DAYDATA'
//ØEXECØCAPTURE,DSN='W.EDS.CCEA.DAILY'
//FT05F001ØDDØ*
001     (eg., for January)
/*
```

To CAPTURE data which spans a two-month period, the following input is used:

```
//ØEXECØCAPTURE,DSN='CCEA.JAN.SRTD',UNIT=TAPE9,VOL=Exxxxx
//FT05F001ØDDØ*
001
/*
//ØEXECØCAPTURE,DSN='W.EDS.CCEA.DAILY'
//FT05F001ØDDØ*
032
/*
//ØEXECØCAPTURE,DSN='CCEA.DAILY.JAN',UNIT=TAPE9,VOL=Exxxxx
//ØEXECØCAPTURE,DSN='W.EDS.CCEA.DAYDATA'
/*
```

B. GROUP II (U.S.S.R. and Canada)

When running Group II the CAPTURE program that accesses W.EDS.CCEA.DAYDATA is on the CAPTURE procedure card in the JCL set-up; therefore the EXEC card need only read: //ØEXECØPGM=CAPTURE. See examples below.

To CAPTURE data entirely within the current month, the following input is used:

```
//bEXECbbPGM=CAPTURE
//bEXECbCAPTURE,DSN='W.EDS.CCEA.DAILY'
//FT05F001bDDb*
001    (eg., for January)
/*
```

To CAPTURE data which spans a two-month period, the following input is used:

```
//bEXECbCAPTURE,DSN='CCEA.JAN.SRTD',UNIT=TAPE9,VOL=Exxxxx
//FT05F001bDDb*
001    (eg., for January as a past month)
/*
//bEXECbCAPTURE,DSN='W.EDS.CCEA.DAILY'
//FT05F001bDDb*
032    (for February)
/*
//bEXECbCAPTURE,DSN='CCEA.DAILY.JAN',UNIT=TAPE9,VOL=Exxxxx
//bEXECbPGM=CAPTURE
/*
```

C. Spanning two years

Two runs of the CAPTURE program are needed; one will capture data for the end of the first year and the other will capture data for the beginning of the new year.

When CAPTURE is used on North American datasets for the end of the first year, the Julian date for December 1 is input as it was for previous December runs. Nothing else is required.

When CAPTURE is used on North American datasets for the beginning of the new year, the Julian date for January 1, 001, is input as expected. On the next input card the value of the new year is punched:

```
//bEXECbCAPTURE,DSN='appropriate name'
//FT05F001bDDb*
335
/*
//bEXECbCAPTURE,DSN='appropriate name'
//FT05F001bDDb*
001
1977
/*
```

When CAPTURE is used on non-North American datasets, no input

is needed on either run because both the year and the Julian date can be read from the dataset.

In all cases when two years are spanned, the year is needed to determine if the first year is a leap year so that the Julian dates greater than 365 can be adjusted correctly.

4.2.2.3 EDITING FILE CROPDATA (EDIT)

The EDITOR program is called by the procedure EDIT, and checks for inverted temperatures or missing data. There are three checks and two edits in this procedure. EDITOR scans the data twice and replaces or corrects missing data either by using data from the three closest stations or statistics generated by the primary station to find a substitute. The third time EDITOR checks the data and prints it out indicating the stations it could not correct, and stations where there are days with an unusually large or small gap between the MX and MN. The input for both GROUP I and GROUP II is as follows:

```
//ØEXECØEDIT
//FT05F001ØDDØ*
ØØØ0ØØØ1ØØØ1ØØØ0
ØØØ0ØØØ1ØØØ1ØØØ0
ØØØ0ØØØ2ØØØ0ØØØ1
/*
```

The EDIT PROC needs 250K.

To correct the data through input, use the INSERT procedure, explained in Section 4.2.2.5

4.2.2.4 RUNNING THE ROBERTSON PHENOLOGICAL MODEL (SUZYQ)

SUZYQ is the Crop Calendar procedure. It executes program CROPCALN which uses the daily temperatures to compute how much the crop should have progressed each day for the 14-day period. SUZYQ prints out to three files: GRIDINFO, which contains the daily increments of the crop stage and is subsequently put onto tape; ARNO, which allows the summary data (the last day of the period) to be accessed from a remote terminal in Houston: and MASTER, which has two files, 1 and 2, that writes the daily increments to the "old" file and the stage of the 14th day to the "new" file. The input is as follows:

```
    //ØEXECØSUZYQ,OLD=1,NEW=2,DISP=OLD
or  //ØEXECØSUZYQ,OLD=2,NEW=1,DISP=OLD
```

A record must be kept of which Master file (1 or 2) is used for input and output for each period (i.e., for OLD and NEW). The 14th (or last) day of the period is written onto the "new" file each run, which on the next run becomes the "old" file. The Crop Calendar procedure will begin incrementing at the correct date, using the 14th day of the "old" file to calculate the new starting date. The Master files are reversed each time SUZYQ is run. The SUZYQ procedure produces a summary output which prints out the last day of the period and the crop's stage.

It is easiest and most expedient to run EDIT and SUZYQ together. The EDIT "checker" Subroutine prints MX and MN, the difference between them, and indicates the days containing inverted temperatures or a wide or small gap between MX and MN by an 'XX'. SUZYQ prints the stations that have inverted temperatures and the day(s)on which this occurs (day 1, day 2, etc) in the 14-day period. These error messages appear in summary form as part of the crop calendar output. The input would be:

```
//ØEXECØEDIT
//FT05F001ØDDØ*
ØØØ0ØØØ1ØØØ1ØØØ0
ØØØ0ØØØ1ØØØ1ØØØ0
ØØØ0ØØØ2ØØØ0ØØØ1
/*                  (2)    (1)
//ØEXECØSUZYQ,OLD=1,NEW=2,DISP=OLD
/*
```

When the two-week period spans two years, regardless of country, the value of the first year must be input:

```
                    (2)    (1)
//ØEXECØSUZYQ,OLD=1,NEW=2,DISP=OLD
//FT05F001ØDDØ*
1976
/*
```

As in the CAPTURE routine, the year is needed to determine if the first year is a leap year, so Julian dates in January can be adjusted correctly.

4.2.2.5 MANUAL EDITING OF FILE CROPDATA (INSERT)

If corrections need to be made for inverted temperatures or missing data, they are made with the INSERT procedure, which calls the program INSDEL. INSDEL will both insert and delete information; the two options are discussed separately.

A. To insert data.

```
//bEXECbINSERT
//FT05F001bDDb*
INSbXXXXXbmxbmnb000b000
aaaaabbbbbccccddddd....
/*
```

where: XXXXX is the station number
mx,mn indicates whether the maximum or minimum temperature is to be changed
000 the range of days, by Julian date, to be changed. If only one day is changed, the second field may be omitted.
aaaaa.. the actual change. in fields of 5 (14I5)

An example -- changing the maximum temperature for station 43758 to 61 and 63 on days 79 and 80 respectively:

```
INSb43758bMXb079b080
bbb61bbb63
```

If only one day for a station is changed, the input would be:

```
INSb43758bMXb079
bbb61
```

Any number of INSERT change cards may be run at the same time, in groups of two as shown in the above examples.

When the INSERT procedure is run, SUZYQ should be run as a separate step after INSERT to be sure the changes were made and SUZYQ calculated the increments for those stations. One way to make a quick check is to make sure the last day of the period is the same Julian date printed out on the SUZYQ crop calendar summary output. The INSERT program also indicates if the correction was successful by messages routed to the printer.

B. To delete data.

```
//bEXECbINSERT
//FT05F001bDDb*
DELbXXXXXbmxbmnb000b000
/*
```

where: XXXXX is the station number
mx,mn indicates whether maximum or minimum temperature is to be deleted
000 the range of days, by Julian date to be deleted. If only one day is deleted, the

second field may be omitted.

An example:

```
DEL⌀43758⌀MN⌀083⌀085
DEL⌀43758⌀MX⌀083⌀085
```

This example would replace the maximum and minimum temperature values on days 83 to 85 with '9999' (indicating missing values).

As with the INS option, any number of delete cards may be run at the same time.

4.2.2.6 LISTING OF FILE CROPDATA (LIST)

The LIST procedure calls program PRINT and can be executed at any time to list the contents of the CROPDATA file. In all the procedures that are run for the Crop Calendar (CLEAR, CAPTURE, EDIT, SUZYQ, INSERT) the LIST procedure produces the listing of the data on CROPDATA and should be run with each program.

The input would be: //⌀EXEC⌀LIST

4.2.3 COPYING FILE GRIDINFO TO TAPE (DSJSCTAPE)

This procedure, using program JSCTAPE, is run to put the crop calendar daily increments from file GRIDINFO onto tape to be sent to Houston. The JCL for DJSCTAP1 (for GROUP I) or DJSCTAP2 for (GROUP II) follows:

```
job card
//⌀⌀⌀TIME=1,CLASS=C
//⌀EXEC⌀NPLIXCLG
//PLI.SYSIN⌀DD⌀*
//GO.IN⌀DD⌀DSN=W.EDS.CCEA.{GROUPI }.GRIDINFO,DISP=SHR
                         {GROUPII}
//GO.OUT⌀DD⌀UNIT=TAPE7,DISP=(NEW,KEEP),LABEL=(1,BLP,,OUT),
//⌀⌀⌀DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200,DEN=2,TRTCH=ET),
//⌀⌀⌀VOL=SER=Exxxxx
//GO.SYSIN⌀DD⌀*
label card
/*
```

On each run the VOL=SER='tape number' must be changed and the appropriate date for the period must be put on the label card

following the card //GO.SYSINØDDØ*. The format for this card is as follows:

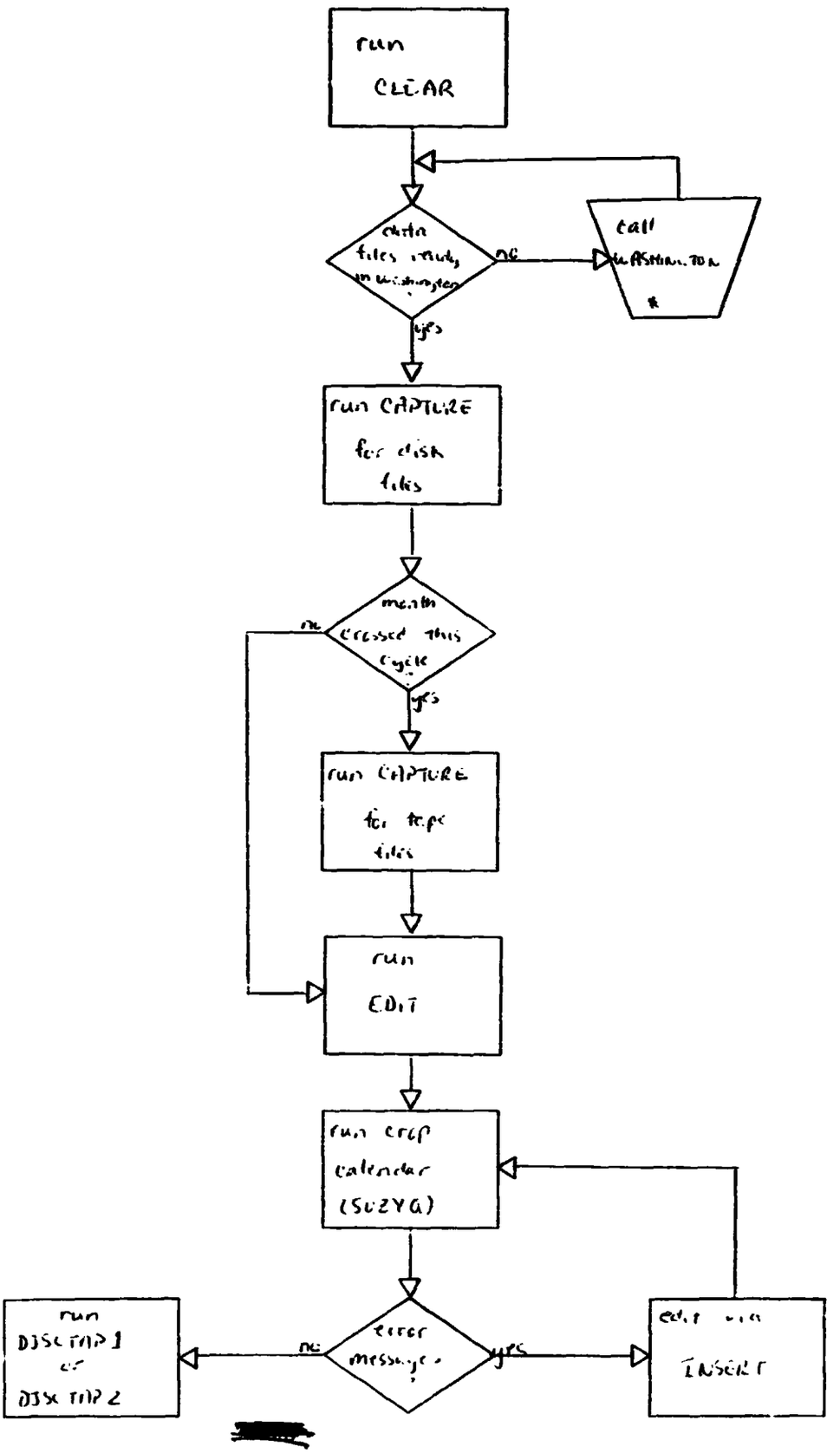
columns 1-6	CCCOUT
7-8	year of first day of the run period
9-11	Julian date of first day of run period
13-14	year of last day of run period
15-17	Julian date of last day of run period
19-80	commentary - used to indicate the GROUP and countries on the tape

example: CCCOUT77101Ø77114ØGROUPII-COUNTRIES:USSR,CANADA*

After the DJSCTAP job has been successfully run, the tape must be sent to Houston. CCEA/Columbia calls Suitland and gives Mr. Pennington the tape number, the group number, and the Julian dates for that run period. This information is for the outside label.

4.3 FLOWCHART

A flowchart showing the sequence of the crop calendar procedures is found on the following page.



*FTS 634-1833; ask for Paul Geeger